UNIVERSIDAD CARLOS III DE MADRID

GRADO EN INGENIERÍA EN INFORMÁTICA

TRABAJO DE FIN DE GRADO

Desarrollo de una aplicación de cifrado de imágenes en el sistema Android.

Autor: Iñaki Rodríguez López

Tutores: Miguel Ángel Patricio Guisado y Antonio Berlanga de Jesús

"As predicted years ago, that, that was, that is, that is no more"

Dave Mustaine.

"La sombra es un color como lo es la luz, pero menos brillante. La luz y la sombra son sólo la relación de dos tonos."

Paul Cézanne.

"If you want to accomplish something in the world, idealism is not enough - you need to choose a method that works to achieve the goal."

Richard Stallman.

AGRADECIMIENTOS

A mis padres, Iñaki y Maribel. No solo por haberme costeado la universidad sino por su incesante apoyo y confianza en mí. También a mis hermanos Óscar y Jorge, que me han marcado un punto desde el cual evolucionar, así como estar presentes en casi todos los momentos en los que necesitaba desconectar.

A mi abuela Trini, por su confianza, devoción y lecciones de cocina. También a mis tíos Merce, Luis, Adolfo y Raúl que siempre me apoyaron en mi toma de decisiones y me empujaron hacia adelante cuando lo he necesitado.

A mis tutores Miguel Ángel y Antonio por su forma incansable de asesorarme y guiarme a lo largo de todo el proyecto.

A toda la "frikitonga": A Sergio por sus evidentes exigencias en seguridad. A Kike por hacerme ver lo cerca que estábamos del final de la carrera. A Nico por manipularme para hacer trabajo de más. A Charlie por aguantar mis quejas, mantener mis pies en el suelo y enfatizar mi pequeño lado bueno.

A mis amigos. A Kris y Mery porque son capaces de haberme mantenido a raya, confiado en mí, haberme escuchado cuando lo he necesitado y haberme hecho ver mis enormes errores, eso sin dejar de mostrarme el más fuerte de los apoyos. A Chovix por sus intervenciones demasiado alejadas en el tiempo y demasiado necesarias. A la señorita Wayne por esas interminables conversaciones telefónicas de quinceañera. A Gonzalo por su "goneismo". A Koa por su amor incondicional a los videojuegos. A Lucildilla, porque los dos años de "jarringui" no se olvidan ni con terapia.

A Gumpus, porque la vida es música.

A todas esas personas que en menor medida han contribuido al avance y finalización de mi carrera académica, amigos, familiares o compañeros de trabajo, sobre todo a aquellos que me han hecho ver que "no se nada" y que todo en esta vida es aprender. A todas ellas que han contribuido en una evolución positiva de mi persona en estos años.

A todas aquellas personas que se han interpuesto en mi camino pero no han logrado que enterrara el hacha y dejara de luchar. También a ellas por haberme hecho ver errores que he cometido y reaccionar a tiempo para cambiar a mejor. A todas ellas, que aunque caigan en un profundo olvido me han hecho ser en cierto modo mejor "superviviente".

A Indrak y Kinit.

Finalmente, al más importante. A mi tío Tete. Fue él quien me dio acceso a mi primer ordenador, quien me alentaba en mis primeros estudios de la mejor de las maneras y, desde que no está es a quien más echo de menos. Nunca supe cómo de bueno era como programador, pero sé que como persona no podré más que estar en su sombra. Nunca serás olvidado.

RESUMEN

El presente proyecto consiste en un encriptador de imágenes para smartphones con sistema operativo Android. La aplicación permitirá a un usuario encriptar una foto almacenada en su terminal de manera que sólo pueda ser desencriptada, y por tanto legible, si se cumple una seria de condiciones previamente establecidas.

Las nuevas tecnologías están cambiando la manera en la que nos comunicamos. Sin embargo, las aplicaciones de mensajería instantánea, que son las más usadas para este tipo de comunicaciones, ven su seguridad comprometida en varios aspectos. Recientemente, aplicaciones como Whatsapp, Line o Spotbros están reaccionando ante tan preocupantes fallos. Se centran así en la seguridad de los mensajes en texto plano que se intercambian los usuarios. Pero ¿Qué ocurre con las imágenes que éstos intercambian?

En este tipo de aplicaciones, las imágenes, así como demás elementos multimedia, se comprimen y se cifran únicamente en el proceso de envío. Esto quiere decir que, cuando una imagen se envía utilizando estos medios, ve afectada su calidad.

Asimismo, las imágenes almacenadas en el teléfono son visibles para cualquiera que acceda a él. Esto supone un problema de cierta importancia en caso de robo o incluso en el caso de que una persona se haga con nuestro terminal durante un par de minutos y por satisfacer su curiosidad, se dedique a ver las imágenes del teléfono.

El problema no solo reside en fotos con mayor o menor grado de privacidad. Muchos usuarios, utilizan capturas de pantalla del teléfono o hacer fotos con la cámara de éste para almacenar datos, entre los que se encuentran, emails, nombres de usuario o incluso contraseñas.

La seguridad nunca es total. Por ello se considera que, por muy complejo que sea el patrón o contraseña de bloqueo, por mucho que se intenten ocultar las imágenes en una ruta de carpetas ridículamente grande, o hagamos que la galería de imágenes no detecte las “imágenes sensibles”, Siempre existe un modo de acceder a esta información.

Por ello, la aplicación que se va a describir en la presente memoria, propone cifrar las imágenes almacenadas en dispositivos móviles. De esta manera, se pretende que solo usuarios autorizados sean capaces de ver las imágenes que queremos proteger.

La idea también puede aplicarse en contextos comerciales. Si al cifrado de la imagen, le añadimos condiciones, como por ejemplo que únicamente sea visible al conectar el dispositivo a un determinado punto WI-FI, o a partir de cierta hora, podemos utilizar la aplicación para sorteos, rifas, cupones de descuento etc.

Como conclusión, vemos que el presente proyecto viene motivado por dos ideas. La falta de privacidad en las imágenes almacenadas en los smartphones y la utilidad de los propios dispositivos con una aplicación instalada para los usos comerciales que se han mencionado anteriormente.

ABSTRACT

This project consists in an image cipher for Android smartphones. This application will allow the user to encrypt an image which is in their terminal so that it can only be decrypted if a number of conditions are met.

New technologies are changing the way in which we communicate. Nevertheless, instant messaging apps, which are the most used apps for communicating, can have their security jeopardized in many ways. Recently applications such as Whatsapp, Line or Spotbros are reacting to so worrying errors. They focus on messages security of plain text but ¿What happens with the exchanged images?

In these apps, images and other multimedia elements, are compressed and encrypted only in the sending process. This means that when an image is sent using these media have its quality affected.

Moreover, stored images in the smartphone are visible by anyone with access to it. This means a problem of relative importance in case of theft, loss, or if someone uses our terminal for a minute and to satisfy their curiosity and starts watching the stored images of our device.

It is not only a privacy problem. Many users use screenshots or take photos with their smartphones to store important data like emails, user names or even passwords.

Security is never absolute. Due to this, it is considered that no matter how complex the pattern or password is, nor how many efforts we do to hide images in a ridiculously big folder path, nor the fact that we make our gallery not to detect this “sensitive-information”, there are always ways to access that information.

For this reason, the application that is going to be described in this memoir solves this problem encrypting stored images in smartphones. By doing this, the owner ensures that only authorized users are able to see the protected images.

The idea can also be used in commercial backgrounds. You can add specific conditions to the image encryption (making it visible at a concrete time or under a concrete WI-FI. This can be used in contests, raffles, discount coupons…

In conclusion, we consider that this project is motivated by two ideas. The lack of privacy in stored images in our smartphones and the utility of devices with the application installed for aforementioned commercial purposes.

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

ESTRUCTURA DEL DOCUMENTO

La estructura del presente documento se divide en los siguientes apartados:

- **Introducción:** En el que se abordarán las cuestiones referentes al estado del arte y un análisis actual del mismo desde el punto de vista histórico, tecnológico y jurídico.
- **Objetivos:** Donde se describen los objetivos que se van a marcar para el desarrollo de la aplicación.
- **Análisis:** En el que se realizará el Análisis del sistema de información que se va a desarrollar.
- **Diseño:** En el que se realizará, con el producto del Análisis, un proceso de Diseño más detallado del sistema de la información que se va a desarrollar.
- **Implementación:** Donde se detallará cómo deberá llevarse a cabo la implementación de la aplicación que se va a desarrollar.
- **Evaluación y resultados:** Que consistirá en una serie de pruebas para comprobar, y corregir en caso de que fuese necesario, el funcionamiento de la aplicación. Además se llevará a cabo un estudio de rendimiento.
- **Conclusión:** Donde, en vista de los objetivos alcanzados, se realizará una evaluación y se propondrán trabajos futuros sobre la aplicación ya desarrollada.
- **Referencias:** Que enumerarán las referencias necesarias para el desarrollo del documento.
- **Anexos:** Que constan de un pequeño diccionario de términos y acrónimos, manual de usuario de la aplicación, posibles usos de utilidad de la aplicación, así como el presupuesto del desarrollo de la misma.

1. INTRODUCCIÓN

1.2. CONTEXTO ACTUAL

Desde que el hombre tiene uso de razón, siempre ha tratado de plasmar lo que ve sobre medios materiales. Desde las pinturas rupestres, las imágenes han ido cambiando ese medio. La piedra de una cueva, un bajo relieve en madera, el lienzo, el papel, la fotografía… Y aunque en principio las imágenes han sido el principal impulsor del arte visual, no siempre se quiere enseñar una imagen de manera pública.

Siempre que ha habido medios, ha habido ciertas imágenes, planos de maquinaria de guerra, mapas del tesoro, pistas que siendo un recursos gráficos se han querido mantener en secreto. Esto no ha cambiado con el paso del tiempo. Lo que sí está en constante cambio, es el propio material sobre el que se plasma la imagen. Se ha hablado de papel o lienzo, pero con la aparición de nuevas tecnologías, como los computadores o internet, las imágenes han dejado de ser prácticamente físicas, para convertirse casi en su totalidad en datos, que ordenados de una manera concreta forman una imagen. El soporte físico no desaparece, sino que evoluciona. La imagen siempre tiene que ser visible. Antes era en una pintura o en un cuaderno, ahora es en una pantalla.

Es aquí cuando recordamos el problema de la privacidad. Los planos secretos, mapas y en general, fotos que por un motivo u otro son comprometedoras o dañinas para nuestra persona, siguen estando casi siempre al descubierto. Se hace mucho hincapié en proteger la privacidad de nuestras comunicaciones durante el tiempo en el que estamos manteniéndolas, pero, ¿qué ocurre con los datos registrados? ¿Y si esos datos son imágenes?

Además, la aparición de los teléfonos inteligentes plantea otro reto en este sentido, muchos usuarios, utilizan la cámara de su teléfono para almacenar imágenes a modo de recordatorio, desde recetas médicas, contraseñas de entidades bancarias, etc. Es una práctica muy incorrecta pero a la vez muy extendida. Por no hablar de las fotografías que por el motivo que sea, pueden dañar nuestra imagen o la de algún conocido, pero que aun así se almacenan en el teléfono.

Es alarmante la frecuencia con la que esto ocurre. Podemos añadir un aliciente más. Los teléfonos inteligentes son uno de los productos más consumidos de los últimos años. Por tanto son muy valorados. El robo de teléfonos móviles se incrementa de manera exponencial. ¿Qué ocurriría entonces con esos valiosos datos?.

Los datos sensibles no solo pueden ser leídos por ladrones. Siempre encontraremos un amigo o familiar que sepa el código o patrón de desbloqueo de nuestro teléfono, con lo cual esos datos también quedarían expuestos.

Por ello es conveniente cifrar los datos. Centrándonos en las imágenes, vemos que no existen muchas maneras de encriptarlas en el mercado. Existen muy pocas aplicaciones que encripten ficheros y nunca trata los datos como imágenes. Entonces ¿cómo puedo hacer que una determinada imagen en mi smartphone la puedan visualizar uno o unos pocos usuarios? ¿Qué pasa con las imágenes sensibles en caso de robo del teléfono?

La respuesta a esas preguntas es la aplicación que se va a desarrollar.

1.2. ESTADO DEL ARTE

El desarrollo de la aplicación necesita del conocimiento de varios aspectos. No sólo eso, se deben tener en cuenta la evolución que están teniendo y que pueden tener en un futuro próximo. Por ello, se ha decidido dividir el estado del arte en varios aspectos generales.

1.2.1. SMARTPHONE: TELÉFONO INTELIGENTE

Un smartphone o teléfono inteligente, se define como un teléfono móvil, que basado en un sistema operativo definido específicamente para este fin, es capaz de realizar gran número de tareas. El concepto surge de juntar los teléfonos móviles "tradicionales" con los dispositivos conocidos como PDA (Personal Digital Assistant). Mientras que un teléfono móvil tradicional únicamente podía cubrir las funciones de realizar llamadas, y enviar mensajes cortos o multimedia, los más avanzados incluyen una cámara de fotos, la PDA, está más orientada a manejar la información de carácter personal. Las PDAs, solían incorporar conexión inalámbrica WI-FI, eran capaces de acceder a internet mediante navegadores, enviar correos electrónicos e incorporaban la por aquel entonces novedoso sistema de pantalla táctil con bolígrafo óptico. Algunas de ellas combinaban la pantalla táctil y un pequeño teclado físico. Con el paso del tiempo, ambos dispositivos evolucionaban incorporando mejoras y funcionalidades, hasta el punto de que comenzaron a incorporar las mismas. No existe una diferencia definida o establecida entre un móvil tradicional y un smartphone, aunque el hecho de que los smartphones utilicen interfaces de programación de aplicaciones (APIs) para dar soporte a software de terceros es la diferencia más significativa [10].

ILUSTRACIÓN 1: ERICSSON GS88

El concepto de smartphone se acuña en el año 1994 por la empresa Ericsson, que definió así su modelo GS88 con fines publicitarios. A finales de los años 90, la mayoría de los teléfonos tenían un pequeño soporte de lenguaje Java, dado que el mercado de la telefonía se publicitaba mucho con juegos de pequeño coste programados en este lenguaje. Las aplicaciones en Java no podían desbordar los recursos, muy escasos, del teléfono. Por otro

lado las PDAs ya iban incorporando sistemas operativos más estandarizados como Blackberry OS o Windows Pocket PC [11].

En los años posteriores, por sencillas razones de demanda del mercado, se empezaron a incluir sistemas operativos específicamente diseñados para los terminales móviles. Los sistemas operativos móviles, incluyen un conjunto de servicios básicos para el funcionamiento del smartphone, es decir, redes celulares, navegación GPS, soporte para WI-FI, cámara de fotos y video, reproductor de archivos multimedia, pantalla táctil…

Recientemente se van incorporando nuevas tecnologías que tienen menos importancia. Por ejemplo la versión de Android más reciente, 4.4 Kit-Kat, se centra en mejorar el uso de la multitarea o centralizar los mensajes (sms, email, chat,…), o soporte para conectividad 4G, que está basada completamente en IP y permite una velocidad mucho mayor [12].

Asimismo, los terminales también evolucionan a nivel de hardware. Hoy en día los teléfonos tienen una pantalla capaz de reproducir multimedia en una resolución de 1920x1080 a 445 ppp, mayor que la que es capaz de reproducir un monitor de hace unos años. Cuentan con una cámara de 8 megapíxeles, más potentes que las cámaras digitales de gama media, con reconocimiento facial y zoom óptico. El rendimiento de muchos teléfonos es mucho mayor que el de muchos portátiles de una antigüedad de unos 5 años. Incluyen procesadores de hasta 2,26 GHz, con procesamiento de gráficos independiente y 2GB de RAM. Los smartphones más novedosos cuentan con reconocimiento de huella dactilar (Iphone 5s) o recarga de batería inalámbrica (Google Nexus 5) [13].

ILUSTRACIÓN 2: IPHONE 5S

Si centramos la vista en el mercado de aplicaciones, encontramos que cada sistema operativo actual, cuenta con una plataforma que las aloja. No solo eso, también restringe aplicaciones potencialmente peligrosas, realiza las funciones de intermediario entre el consumidor final de la aplicación con el desarrollador, realiza la compra-venta, mantiene un sistema de puntuaciones y valoraciones de usuarios, etc. Cada una de esas plataformas, Google Play, Apple Store, etc. Tiene sus condiciones para el desarrollo y sus restricciones de seguridad. Por ello, se ha decidido incluir estos aspectos, que han condicionado en gran medida la elección de un sistema operativo en concreto para el desarrollo de este proyecto, en los siguientes apartados, donde también se valoran otros aspectos,

específicos de cada plataforma. Para la realización de este proyecto, se han analizado los dos sistemas operativos móviles que más mercado abarcan: Google Android y Apple iOS.

GOOGLE ANDROID

Android fue una compañía creada en 2003 en Palo Alto, California que fue comprada por Google en 2005. La primera versión del sistema operativo salió al mercado en 2008. Esta primera versión incluía funcionalidades como ventana de notificaciones desplegable, "widgets" o integración con el servicio Gmail.

La versión Android 1.5 "Cupcake" fue la primera en ser nombrada como un postre, algo que Android, de manera alfabética hace desde entonces. La segunda versión introducía las novedades de teclado virtual y captura de fotos y videos.

El cambio más notable en la versión de Android, llegó con Android 2.0 "Eclair". Incorporaba soporte para varias cuentas de Google, navegación GPS, soporte HTML5 para el navegador integrado, reconocimiento de voz y una innovadora pantalla de bloqueo. Android 2.2 "Froyo" mejoraba notablemente el rendimiento y la velocidad. Android 2.3 "Gingerbread" facilitó mucho el desarrollo de aplicaciones, especialmente los juegos.

La versión 3.0 "Honeycomb", está diseñada específicamente para tablets. Cuenta con integración de botones virtuales para paliar la ausencia de botones de estos dispositivos. También se rediseñaron multitud de aplicaciones.

La versión 4.0 "Ice-cream Sandwitch" se lanza en 2011. Supone un importante salto en lo referente a rendimiento. Se mejoró el teclado, el corrector de palabras mal escritas e incorpora la función de desbloqueo facial. Varias versiones sucedieron a Ice-cream Sandwitch, la más actual es Android 4.4, "Kit-Kat", que, aun poniendo especial atención al rendimiento, también incorpora muchas mejoras gráficas, de integración de multimedia, centralización de contactos, posibilidad de gestionar o imprimir documentos de manera directa (sin pasar por un computador "tradicional"), así como muchos otros cambios menores. [14]

La mayor fortaleza de Android es su mercado. Hoy en día es el sistema operativo móvil más extendido. Asimismo, el mercado de aplicaciones también es mucho mayor que el de sus competidores. En Marzo de 2009, había unas 2.300 aplicaciones disponibles. En nueve meses, se multiplicó el número por siete. En Mayo de 2013, se han registrado un total de 9 billones de aplicaciones en el Android Market, la plataforma de descargas. [15]

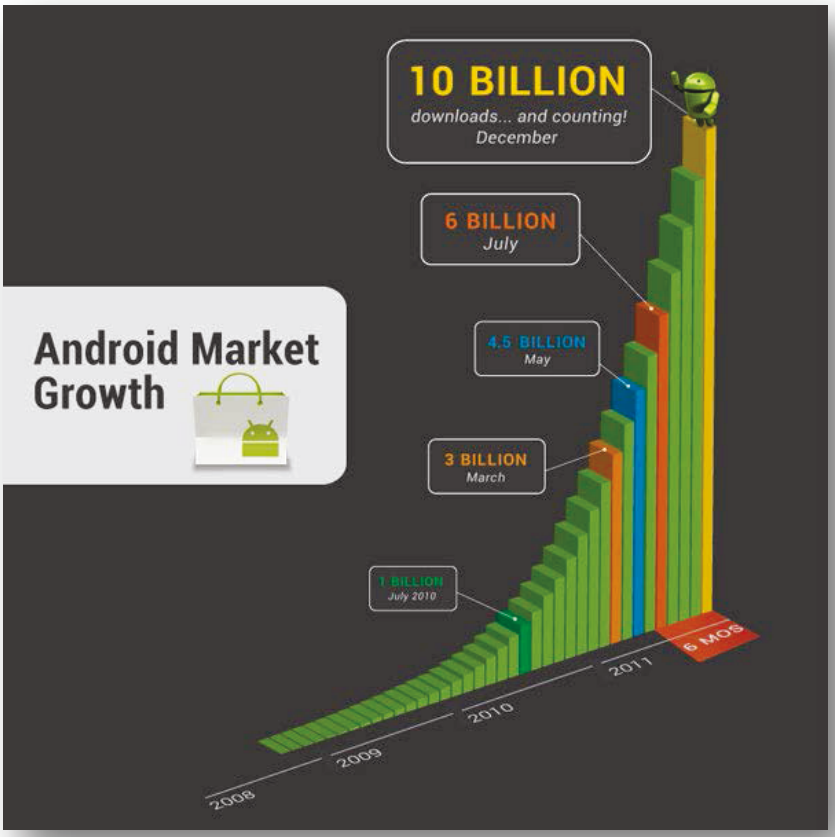


ILUSTRACIÓN 3: INCREMENTO DEL NÚMERO DE APLICACIONES DE ANDROID

El crecimiento tan desorbitado del mercado de aplicaciones, reside en la facilidad de desarrollo. El código del propio sistema operativo es abierto casi en su totalidad, liberado bajo licencia Apache 2.0. Además está basado en un Kernel de Linux. Esto implica que desarrollar una aplicación y subirla en el Android Market es gratuito.

Las herramientas de desarrollo son muy sencillas. Únicamente es necesario descargar el SDK de Android de su página oficial. El SDK contiene un entorno de programación basado en el conocido Eclipse, así como, entre otras, las siguientes herramientas:

- Emulador Android: Que permite ejecutar y depurar aplicaciones seleccionando la versión y el dispositivo.
- Dalvik Debug Monitor Service: Herramienta para depurar, detener procesos, y obtener información general.
- Android Debug Bride: Consola de comandos para comunicarse con instancias del emulador, o del dispositivo Android sobre el que se esté depurando.
- Android Interface Definition Language: Que permite definir la interfaz comunicación entre procesos o servicios.
- SQLite: Que provee acceso a archivos creados o gestionados por las aplicaciones de Android.
- MkSdCard: Permite la creación de tarjetas SD virtuales.

Si queremos tener en cuenta la complejidad de un futuro desarrollo, debemos mirar la documentación disponible. Al tratarse de un sistema tan abierto, existen multitud de tutoriales, desde el más sencillo “Hola Mundo”, hasta tutoriales mucho más complejos.

Cabe destacar la existencia de numerosos video-tutoriales. Además, suelen venir acompañados se códigos de ejemplo, ya que se trata de un sistema operativo sin restricciones de licencia. El propio Android SDK, viene con una extensa documentación y ejemplos de desarrollo de aplicaciones básicas. Ante un problema, existe una numerosa comunidad muy activa dispuesta a resolver dudas.

En cuanto a una previsión en la curva de aprendizaje, hay que destacar que el desarrollo de aplicaciones en Android es muy similar al desarrollo en Java. Independientemente de lo relativamente sencillo que es aprender a programar en un lenguaje de tan alto nivel, hay que sumar la documentación, ejemplos de código y capacidad de resolución de problemas de este lenguaje. Un desarrollador Java no debería tener demasiados problemas en aprender a crear aplicaciones en Android.

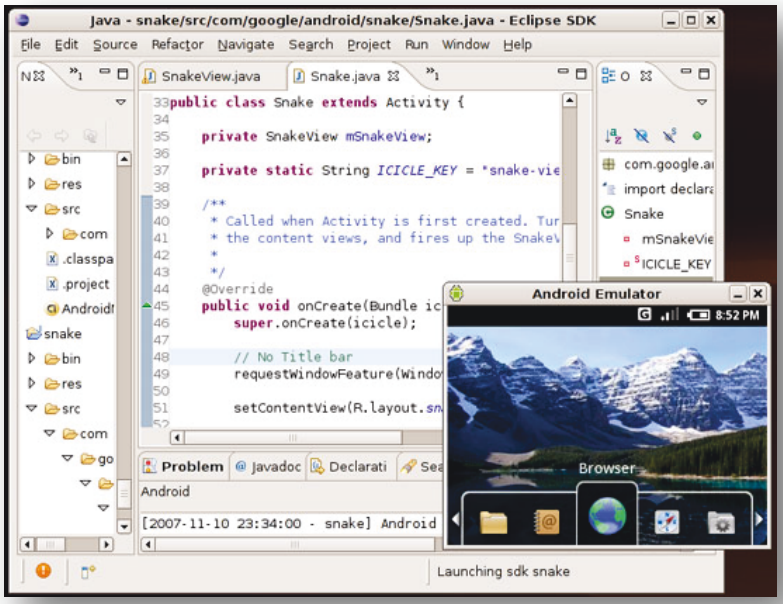


ILUSTRACIÓN 4: ANDROID SDK

Android tiene, no obstante, riesgos implícitos que son importantes desde el punto de vista de la seguridad. La principal es que acepta aplicaciones de repositorios no oficiales. Esto puede ser una ventaja en ciertos casos, como la posibilidad de que una empresa tenga sus propias aplicaciones privadas, desarrolladas para su uso por parte de sus trabajadores y que no estén disponibles para personas ajenas a esa empresa. Además Android tiene la posibilidad de instalar ROMS (firmware) que mejoren el rendimiento del teléfono. Existen multitud de ellas, con varias versiones y pueden contener mejoras de gráficos, sonido, ciertas aplicaciones, etc.

El hecho de que puedan instalarse aplicaciones no oficiales también puede ser una desventaja si tenemos en cuenta todo el malware que puede aparecer en estos repositorios. La instalación de aplicaciones no oficiales siempre conlleva cierto riesgo y no es muy recomendable. Solo la posibilidad de que se ejecute un instalador en segundo plano (posible, puesto que Android es multitarea, es decir, puede realizar más de una función a la vez) y se instale una aplicación maliciosa, va a ver comprometida la seguridad del terminal.

Por suerte, otra de las propiedades de Android es la existencia de Sandboxing. Sandboxing es un término para definir el "aislamiento de procesos". Permite la ejecución de varios procesos de manera paralela, pero sin que uno interfiera en el otro. El ejemplo más obvio en este caso, es el uso de sandboxing para evitar que una aplicación de dudosa fiabilidad interfiera con el propio sistema operativo o con procesos de otras aplicaciones. No sólo eso, también permite limitar los recursos que se consumen con cada proceso.

También hay que resaltar que las aplicaciones necesitan que se les conceda permisos antes de ser instaladas. En otras palabras, el usuario acepta que una nueva aplicación vaya a utilizar uno o más servicios del teléfono. Estos servicios pueden ser, envío de emails, sms, realización de llamadas, conexión a internet, acceso a datos de la tarjeta SD, o incluso a los contactos de la agenda. El problema viene cuando esos permisos no pueden modificarse. Los permisos se deben aceptar todos o ninguno. Sería muy interesante poder elegir que una aplicación, por ejemplo, pueda conectarse a interne pero no realizar llamadas con el fin de ahorrar costes en la factura del teléfono. Sin embargo eso no es posible en Android. Normalmente un usuario acepta los permisos sin apenas mirarlos, lo que, aun siendo responsabilidad última del usuario, también supone una bajada en la seguridad de este sistema operativo móvil.

Otro aspecto que es interesante analizar de Android es que no necesita un software que haga de intermediario entre el dispositivo y un equipo. Por ejemplo, para transferir ficheros de cualquier tipo de un teléfono Android a un PC, independientemente del sistema operativo (aunque al interactuar con MAC-OS tiene muchos fallos), sólo tenemos que conectarlo al equipo.

APPLE IOS

iOS es un sistema operativo diseñado por Apple inc. para sus dispositivos móviles como iPod, iPhone o iPad. iOS está basado en una variante de kernel de Mac OS X, el sistema operativo de los computadores de sobremesa de la propia empresa Apple [16].

La existencia del sistema iOS fue revelada en la Macworld Conference en enero de 2007 bajo el nombre de iPhone OS. Hasta entonces, se supuso que el esperado teléfono de Apple funcionaria con la versión de Mac OS X de aquel entonces. Había gran expectación también por la aparición del iPod Touch, un dispositivo táctil que cumpliría las características multimedia de un iPhone pero sin la posibilidad de realizar llamadas.

En enero de 2010 se anuncia el primer iPad. Un dispositivo muy similar al iPod, pero más enfocado a la industria de contenidos. Una de las principales características era el tamaño de la pantalla, ya que una de los principales motivos para su creación era la posibilidad de leer libros electrónicos en él. iPad también utilizaría el mismo sistema operativo que iPod y iPhone. Debido a que el sistema operativo ya se utilizaba en más de un dispositivo que no era un iPhone, se decidió que el sistema pasase a llamarse simplemente iOS. No obstante, puesto que se trata, al fin y al cabo, del mismo sistema operativo, se hará referencia a él con su nombre actual, iOS.

ILUSTRACIÓN 5: IPAD Y IPHONE CON IOS 6

iOS 1.0, "Alpine" sale a la luz en 2007. Apenas un par de días más tarde, se lanzó la versión 1.0.1 que tenía un parche de seguridad del navegador Safari. Veinte días después, sale la versión 1.0.2, con más correcciones del sistema operativo. Esto demuestra que Apple se toma muy en serio las actualizaciones en sus sistemas.

Se fueron llevando a cabo varias actualizaciones de la primera versión a lo largo del año. Se incluyeron numerosos cambios, como por ejemplo, iconos para accesos rápidos a ciertas funciones, la incorporación de la tienda de música online de Apple, iTunes, salida a televisión, soporte para teclado, incorporación del nivel de batería, así como varios cambios en la seguridad.

En el año 2008, aparece la versión 2.0 "Big Bear", que también incorporaba mejoras, además de un importante cambio en la interfaz y por tanto en la manejabilidad de iOS. Se añaden funciones multi-idioma, videos, búsquedas de contactos, gestión múltiple de sms y emails,… Durante 2009, con la versión 3.0 "Kirkwood", los desarrolladores se centraron en mejorar la seguridad y la integración del sistema con iTunes. Además de la mejora de compatibilidad con la tienda de aplicaciones App Store.

Otro importante cambio llega con la versión de iOS 4.0 "Apex", que se lanzó en 2010. La principal fortaleza de esta versión es la incorporación de las capacidades de multitarea. Hasta el lanzamiento del terminal iPhone 4, los dispositivos Apple no tenían multitarea, es decir, no eran capaces de ejecutar dos funciones al mismo tiempo. Aunque se trata de una notable mejora para aplicaciones e incorporación de funcionalidades que se fueran a incorporar en un futuro, exige que los desarrolladores declaren explícitamente cuando éstas se podrán ejecutar en segundo plano y hay que tener en cuenta que el propio sistema operativo limita mucho las ocasiones de cuando esto puede ocurrir.

La versión 5.0 "Telluride" incluyó notables mejoras en el sistema de notificaciones, mensajes, y además la tecnología denominada iCloud [18]. iCloud el primer servicio de almacenamiento en la nube de Apple. No obstante, el mayor avance en cuanto al desarrollo de aplicaciones para este sistema operativo fue la incorporación de Siri en el iPhone 4S.

Siri es una funcionalidad de iPhone que simula un agente personal. Se trata de una aplicación que reconoce el lenguaje natural expresado por el usuario y mediante un impecable uso de inteligencia artificial, navega a través de las funcionalidades del sistema operativo. Siri es capaz de realizar llamadas, consultar el tiempo a través del navegador, abrir distintas apps, etc. [19].

iOS 6, eliminó ciertas aplicaciones de Google (Google Maps y Youtube), una controvertida decisión ya que Google Maps fue sustituida por Apple Maps, cuyo funcionamiento no era nada óptimo. Ninguna aplicación de Apple podía sustituir a Youtube, por razones de patentes. Sin embargo, las aplicaciones de Google para iOS seguían estando disponibles en el App Store. Además de esto, se concentraba en eliminar latencias y optimizar el rendimiento que no tenían las anteriores versiones.

La versión actual, iOS 7, introdujo cambios bastante pronunciados en la interfaz, así como la aplicación iTunes radio, con sincronización de podcasts y mejoró la capacidad de búsquedas en aplicaciones como iTunes y iCloud. Asimismo, se mejora el rendimiento y las capacidades de multi-tarea. Esta versión se publicó a la vez que el iPhone 5S y iPhone 5C, teléfonos que tienen mayor rendimiento que los terminales anteriores, por ello, se dan casos de cierta latencia en teléfonos antiguos [20].

ILUSTRACIÓN 6: DIFERENCIAS GRÁFICAS ENTRE IOS 6 Y IOS 7

Desde el punto de vista del desarrollo, una persona que pretenda hacer una aplicación para iOS, puede desarrollar la app de manera gratuita, sin embrago, las licencias de Apple son en cierto modo restrictivas en cuanto a la publicación. La publicación en la App Store, que es la única plataforma de aplicaciones que permiten los teléfonos con iOS, requiere el pago de una cierta cantidad de dinero, así como la aprobación de la propia empresa. La razón de esto se detallará más adelante.

Un desarrollador es libre de fijar el precio al que su producto se va a vender en la plataforma de Apple, pero a cambio, la empresa requiere un 30% de los ingresos conseguidos. Las aplicaciones pueden ser gratuitas, pero el desarrollador seguirá teniendo que pagar la licencia de desarrollo y publicación, que oscila alrededor de los 100€ mensuales.

Un desarrollador necesita un código por parte de Apple, una clave de firma. Sin esta clave, el desarrollador no podrá publicar. Esto es una restricción de seguridad muy importante, ya que así Apple es consciente de quien es el desarrollador a la vez que se asegura de que no existan aplicaciones externas a su plataforma.

El principal entorno de desarrollo es el Apple Developer Connection (en adelante ADC). Es como se ha mencionado antes, gratuito, aunque el hecho de que haya que registrarse como desarrollador y por tanto, realizar el pago de la licencia lo hace algo más restrictivo. Existen licencias gratuitas para estudiantes, pero hay que presentar cierta documentación y tiene que ser aprobada por el organismo responsable (universidades o escuelas). Junto a ADC, es aconsejable descargar Corona SDK, para ejecutar las aplicaciones que se están desarrollando en terminales iOS. Esta aplicación no es propiedad de Apple, por lo que sí es gratuita. El entorno de programación se denomina X Code. X Code incluye herramientas de ayuda, un emulador de iOS y herramientas para comprobar el comportamiento y la realización de pruebas [21]. X Code solo es soportado para computadores con sistema operativo Mac OS X.

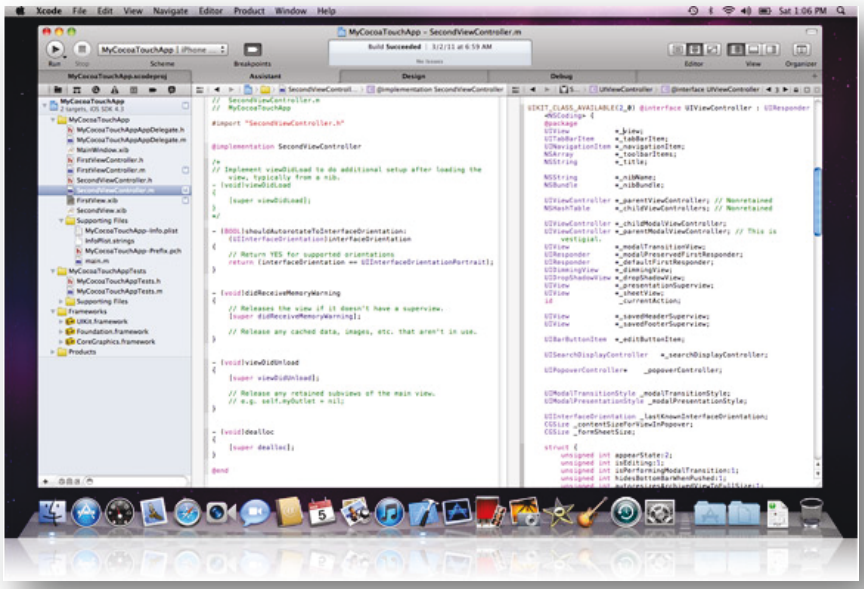


ILUSTRACIÓN 7: ENTORNO X CODE

A nivel de documentación, existen muchos tutoriales online, así como guías y libros específicos para el desarrollo de apps en iOS. La información se encuentra en varios idiomas y de manera sencilla. También hay disponibles muchos ejemplos de código, se encuentran varias aplicaciones sencillas, a modo de ejemplo incluidas en el ADC o X Code. La resolución de problemas es también un aspecto interesante, existen varios foros y blogs [22].

iOS, como lenguaje de programación, tiene una curva de aprendizaje pronunciada al principio, pero mediante la documentación se pueden solventar casi todas las dudas. Un programador que conozca el lenguaje C y la orientación a objetos, C++ o Java, no debería tener problemas para la realización de un trabajo en este sistema operativo.

A primera vista, parece que Apple quiere que el desarrollo de aplicaciones para su sistema operativo sea lo más restrictivo posible. Sin embargo, esto se debe al coste, tanto

económico como de tiempo y personal que conlleva la seguridad del sistema. Aquí se puede introducir el término "Walled Garden" o jardín vallado. Este concepto de plataforma cerrada, choca frontalmente con el de código abierto. La empresa, en este caso Apple, restringe todo lo posible el desarrollo para evitar problemas de piratería (publicación sin derechos de autor) y aplicaciones maliciosas, entre otras. El hecho de que Apple deba aprobar una nueva aplicación para su posterior publicación implica que todas las aplicaciones del App Store, han sido verificadas por la empresa, es decir, tienen la garantía de Apple de que no se trata de una aplicación que vaya a dañar nuestros datos o nuestro teléfono, al mismo tiempo que se está realizando una compra de contenidos pagando derechos de autor. Todo ello conlleva un coste, que fijan entre el desarrollador y la empresa Apple y va a parar al usuario [23].

1.2.2. IMAGEN: DEFINICIÓN Y FORMATO

Una imagen se define como "*Figura, representación, semejanza y apariencia de algo*" [24]. Si profundizamos más en nuestro concepto, se trata de figuras bidimensionales que representan objetos o entidades reales a través de ellas en un soporte físico. Desde que el hombre tiene uso de razón, ha utilizado imágenes con distintos fines, tanto decorativos como explicativos. Desde las pinturas rupestres en las paredes de las cuevas, pasando por cuadros sobre lienzos o madera, hasta llegar al día de hoy, donde el soporte físico es muy avanzado y hasta permite el tratamiento de la propia imagen en tiempo real.

Una definición más tecnológica y actual de imagen es "*un fichero que tiene datos almacenados y si, se leen de manera correcta, muestra una imagen en el sentido tradicional*". Las imágenes desde este punto de vista, son puntos agrupados de una determinada manera. Esos puntos son los definidos por el conjunto de datos que forman la propia imagen y se denominan pixeles. El número de pixeles que tiene la imagen viene dado por la resolución de la imagen.

Dependiendo del formato de la imagen (que no es más que la codificación), cada pixel será representado por uno u otro número de bits o incluso bytes. Por eso, la mayoría de las imágenes que podemos encontrar en nuestros computadores y terminales son sencillamente mapas de bits. Es cierto que existen codificaciones de imágenes que permiten que un fichero tenga un formato vectorial, esto es, en lugar de almacenar pixeles a modo de puntos, almacena la forma y la posición de objetos geométricos, como polígonos, arcos o líneas. De esta manera al ampliar la imagen, no pierde calidad.

ILUSTRACIÓN 8: ICONOS DE IMÁGENES EN WINDOWS

Los mapas de bits, como se ha mencionado antes, dependen de su formato. Todos ellos tienen una cabecera que indica, el tipo de formato, el tamaño del archivo, la altura y anchura, etc. Existen multitud de formatos hoy en día pero los más comunes son los siguientes: [25].

- BMP: Bitmap. Consiste en una sencilla sucesión de pixeles. Tiene una gran capacidad para almacenar mucha información, pero a costa de un tamaño de fichero muy elevado.
- GIF: Graphics Interchange Format. Formato diseñado para comprimir imágenes. Únicamente permite un máximo de 256 colores. La compresión que utiliza es idónea para el almacenamiento de imágenes online. Además, permite la animación de las imágenes.
- JPG / JPEG: Joint Photographic Experts Group. Permite el uso de 16 millones de colores. La compresión de este formato implica la pérdida de calidad de imagen. No obstante la compresión es tal que es altamente rentable la utilización del formato, sobre todo cuando se quieren almacenar online imágenes con una calidad mayor que la que soporta el formato GIF. Además pueden almacenarse imágenes en este formato sin perdidas de calidad, aunque con menor compresión.

ILUSTRACIÓN 9: PÉRDIDA DE CALIDAD EN JPEG

- TIF / TIFF: Tagged Image File Format. Formato muy adecuado para editar una imagen. Almacena un gran número de colores y con una calidad muy alta. Por desgracia este formato almacena los ficheros con el tamaño prácticamente más elevado.
- PNG: Portable Network Graphic. Se trata de un formato reciente, con una tasa de compresión superior a la de GIF. Permite almacenar un número total de colores mucho mayor, pero no tiene animación. Sin embargo, si soporta el color "transparente". La cabecera de este formato es mucho más compleja que la de otros formatos. En ella, se especifican multitud de campos, desde el color del fondo por defecto, el balance de blancos, si contiene texto comprimido o no, etc. Este formato es muy útil si se quieren hacer ampliaciones ya que es muy tolerante a este tipo de acciones si se tienen grandes espacios del mismo color en la imagen. [26].

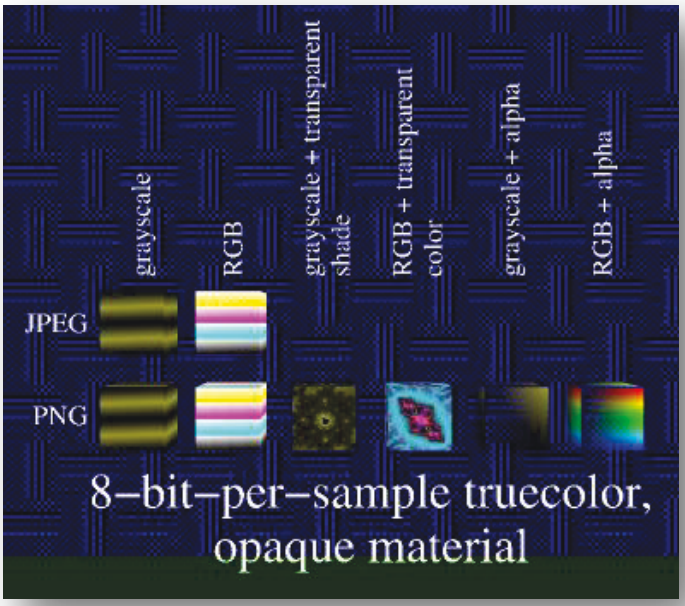


ILUSTRACIÓN 10: COMPARACIÓN DE CABECERAS JPEG Y PNG

Vemos que cada formato se utiliza en función de las manipulaciones que queramos hacer con la imagen. Algunos de ellos, como el GIF, estarían obsoletos de no ser por su funcionalidad única (en este caso la posibilidad de almacenar imágenes animadas). La elección del formato a utilizar dependerá de sendos factores que se detallarán en el apartado de objetivos de la realización del presente proyecto.

1.2.3. SEGURIDAD: CRIPTOGRAFÍA Y CIFRADO

Cifrar se define como *"transcribir en guarismos, letras o símbolos, de acuerdo con una clave, un mensaje cuyo contenido se quiere ocultar"* [1]. Esto es, la alteración de los caracteres de un mensaje de manera que no pueda ser leído por terceros. Un mensaje cifrado con una contraseña, solo será visible por un receptor que conozca la contraseña, o clave, y pueda iniciar un proceso de descifrado para poder leer el mensaje. De esta manera, si el receptor no es el destinatario de lo que queremos transmitir, es decir, es un interceptor, no podrá descifrar ni por tanto leer el mensaje, asumiendo que no conoce la clave mediante la cual éste se ha cifrado [2]. Éste proceso de interceptación de mensajes se describe en la siguiente figura:

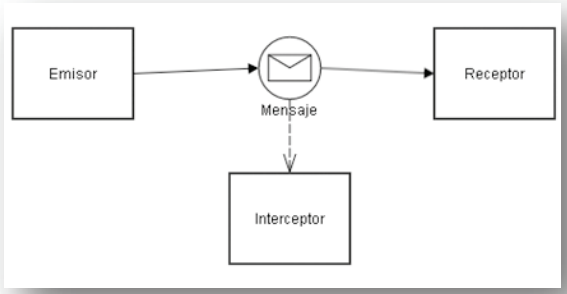


ILUSTRACIÓN 11: INTERCEPTACIÓN DE UN MENSAJE

El cifrado de un mensaje no previene la interceptación de éste. En su lugar previene la lectura del mismo una vez una persona no autorizada se haya hecho con información de manera no lícita.

El cifrado es solo una pequeña parte de la criptografía, el arte de la escritura oculta. La criptografía nació en la antigua Grecia, con lo que se conoce como "criptografía clásica". Se compone de métodos criptográficos de "lápiz y papel" aunque en ocasiones se utilizan mecanismos simples. No es hasta comienzos del siglo XX cuando se empiezan a utilizar maquinas electromecánicas complejas, con la aparición de la Máquina Enigma, el invento más destacado de Alan Turing.

Los primeros datos que se tienen de la existencia de la criptografía datan aproximadamente del año 1900 a.C. que consistían en jeroglíficos egipcios que eran tallados en edificios. No se tratan realmente de mensajes cifrados, pero se consideraron como tal, debido a la imposibilidad de descubrir su significado hasta la aparición de la piedra de Rosetta. No es hasta años más tarde cuando encontramos el cifrado Atbash que usaban los antiguos hebreos. Este sistema primigenio, consistía en un simple cifrado de sustitución, es decir, el intercambio de unos caracteres por otros, de manera que el mensaje no sea legible a simple vista. Los espartanos, alrededor del año 500 a.C. usaban la escítala, un método que consistía en una vara de un grosor determinado y una tira de papiro o cuero donde se escribe el mensaje de tal manera que solo enrollando la tira alrededor de la vara el mensaje es legible.

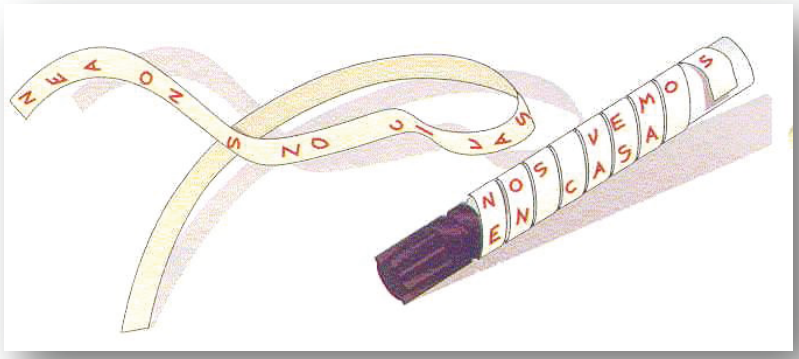


ILUSTRACIÓN 12: FUNCIONAMIENTO DE UNA ESCÍTALA

Sin embargo, los cifrados de transposición, como el que usa la escítala, son muy vulnerables, por ello, los métodos de cifrado utilizados durante buena parte de la historia, consistían en gran proporción en sustitución monoalfabética. Éstos cifrados son ligeramente más sólidos, aunque sensibles a un criptoanálisis bien realizado.

La sustitución monoalfabética consiste, sencillamente en cambiar un carácter por otro del mismo alfabeto. Normalmente se sigue un determinado patrón para fijar un criterio en la sustitución. EL conocido como cifrado César consiste en sustituir un carácter por otro que se encuentre a un número determinado de posiciones de éste en el alfabeto. Es decir, si queremos sustituir la "A" con una clave de tres posiciones, el carácter que debemos incluir para cifrar el mensaje se encuentra a tres posiciones de "A", por tanto, escribiríamos "E". Para descifrar el mensaje, únicamente deberíamos restar ese número de caracteres. [4]

Empezamos pus a hablar del concepto de clave. La clave se define como *"código de signos convenidos para la transmisión de mensajes secretos o privados"* o *"conjunto de reglas y correspondencias que explican este código"* [5]. En los dos ejemplos anteriores, somos capaces de ver ambos conceptos. En el ejemplo de la escítala, encontramos que el conjunto de reglas para poder descifrar el código, es decir, la clave, es el ancho de la vara a la que se enrolla el mensaje. Se trata de una clave única ya que una vara con un ancho distinto al establecido en la clave no serviría para poder hacer legible el mensaje. Con el cifrado César, la clave es el número de posiciones que se va a utilizar para la sustitución. Pero en este caso, podemos dar forma a esa regla y convertirla en una clave escrita, un carácter, el 3. De esta forma, la clave cumple la primera definición, puesto que se trata de un signo convenido para poder cifrar o descifrar mensajes.

Como se ha mencionado con anterioridad, la sustitución monoalfabética es vulnerable a un criptoanálisis. Esto quiere decir, que los mensajes cifrados que eran interceptados, podían ser descifrados sin la clave en un tiempo más o menos razonable. Si por ejemplo se identificaba que se trataba de un cifrado César, solo es cuestión de tiempo, y sin necesidad de mecanismos complejos, dar con la clave y poder descifrar el mensaje interceptado. Éste es el mayor problema en criptografía, los mencionados mecanismos complejos, son en efecto cada vez más complejos y avanzados y siempre se debe idear un cifrado lo suficientemente fuerte como para que el tiempo que se tarde en poder realizar el criptoanálisis y descifrar el mensaje no sea viable.

Los avances en la criptografía clásica fueron pocos. El mayor de ellos fue el uso de alfabetos "extendidos" o varios alfabetos. La sustitución polialfabética, consiste en utilizar caracteres no de un mismo conjunto de ellos, sino de varios. Un claro ejemplo es el ideado por Leon Battista Alberti en 1467. Alberti ideo un sistema de discos móviles para su sistema de cifrado.

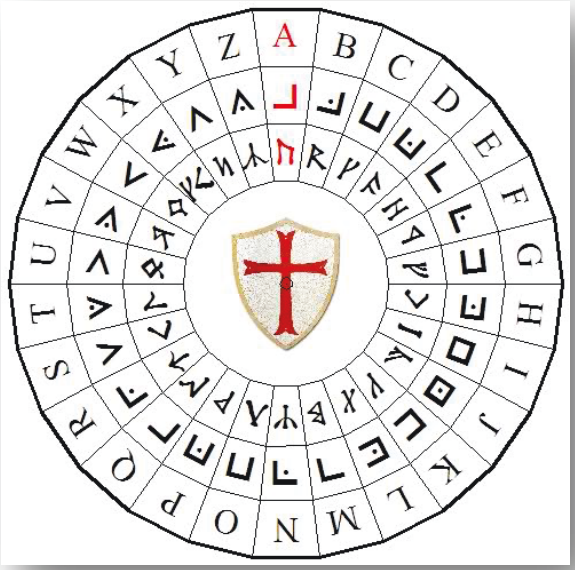


ILUSTRACIÓN 13: LOS DISCOS DE ALBERTI

En este caso, el texto que se cifraba estaba en el alfabeto del disco exterior. Mediante una clave establecida, que era uno de los caracteres de un disco interior el mensaje se cifraba incluyendo caracteres de relleno que indicaban el sentido y la longitud de la rotación de los discos [6]. El novedoso sistema, disparó el tiempo requerido para el criptoanálisis de manera importante.

Lo que más revolucionó la criptografía a lo largo de la historia fue la era de los computadores, ya que se podían encargar de realizar largas operaciones matemáticas en relativamente poco tiempo. Por tanto, la fortaleza del cifrado, ya no se encuentra en la clave, como hemos visto hasta ahora. Con la aparición de los computadores, el punto fuerte de un cifrado se encuentra en el algoritmo que se usa para encriptar el mensaje.

Un claro ejemplo fue la Máquina Enigma, inventada por Alan Turing. Esta máquina, considerada por muchos el primer computador, utilizaba una máquina de estados, una tabla de instrucciones y una cinta de la cual se leía y escribía. Enigma es la primera máquina automática de la historia. La Máquina Enigma fue utilizada por los nazis para encriptar comunicaciones, pero Alan Turing colaboró en la creación del computador Colossus, que descifró importantes emisiones radiofónicas cifradas del ejército alemán en la Segunda Guerra Mundial. El cifrado de estos mensajes, se generaban con un sistema de rotores propulsados por vapor llamado Lorenz SZ40, que utilizaba un cifrado Vernam. El cifrado Vernam consiste en un método sencillo pero efectivo tal y como es realizar una operación lógica de "o exclusivo" (XOR) del mensaje con la clave en base 2. Esto quiere decir, que se pasaban ambos, clave y mensaje, a binario y se realizaba la operación XOR para obtener el resultado cifrado. Para realizar el criptoanálisis de mensajes largos (4.000 caracteres aproximadamente) se requirió una gran capacidad computacional, que resultó ser el primer computador programable [8].

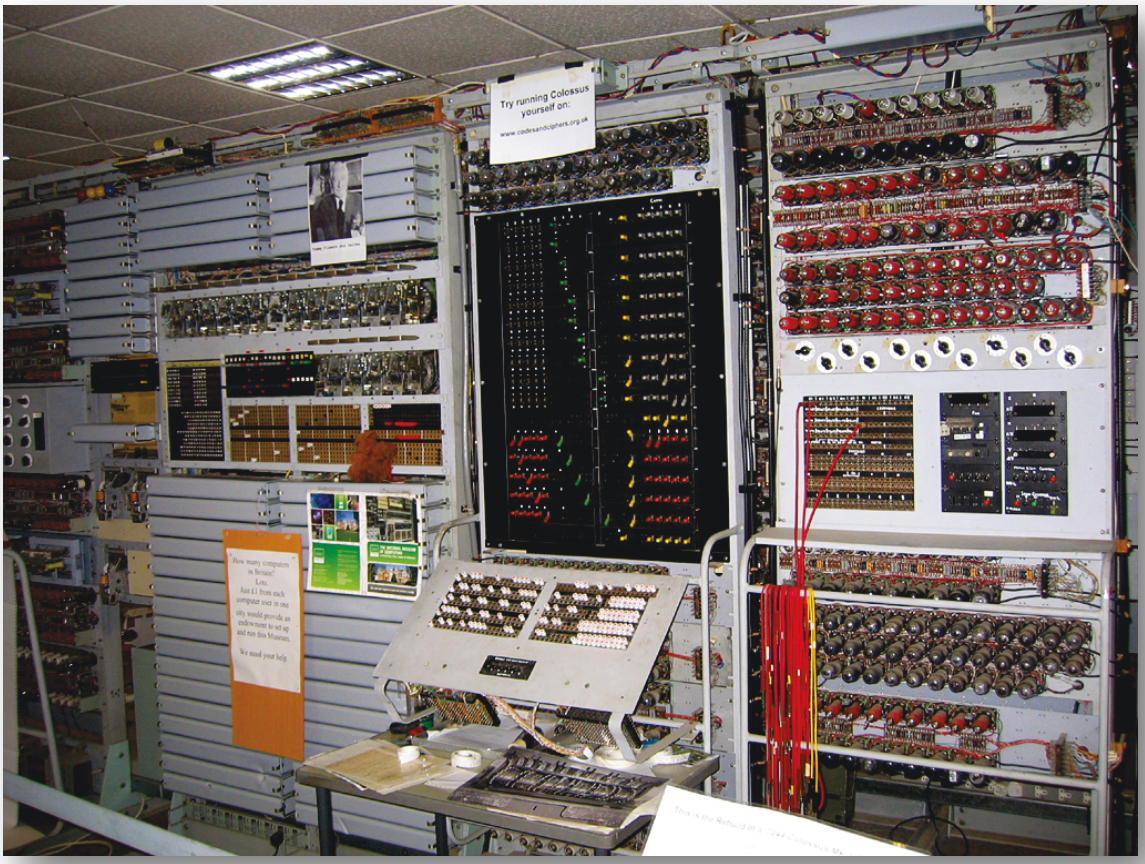


ILUSTRACIÓN 14: EL COMPUTADOR COLOSSUS

Con la aparición de los computadores, la criptografía y la seguridad han estado en constante evolución. Esto se debe a que con el incremento de la capacidad computacional a lo largo de la historia, los cifrados resultan no ser suficientemente fuertes. Antiguamente, un cifrado César podía ser roto en cuestión de días, hoy en día solo nos llevan unas pocas décimas de segundo. Ocurre lo mismo para el cifrado Vernam, un algoritmo de cifrado que en 1944 requería mucha potencia eléctrica, así como un equipo de muy grandes proporciones, se rompe con mucha más facilidad con un equipo de los que estamos acostumbrados a ver en nuestros hogares. Asimismo, la propia innovación tecnológica requiere la evolución de la seguridad. Siempre se están creando utilidades nuevas que necesitan protocolos de seguridad mucho más avanzados que un simple cifrado, primero internet, luego el internet inalámbrico, accesos a sitios web, etc.

Sin embargo, podemos centrarnos en sólo una parte de la criptografía moderna. Los algoritmos de cifrado simétrico. Los algoritmos asimétricos, solucionan el problema del intercambio de claves de la criptografía simétrica, ya que emisor y receptor no tienen que ponerse de acuerdo sobre qué clave utilizar. La criptografía asimétrica busca sobre todo proteger la integridad del mensaje, no su confidencialidad, es por eso que nos centraremos en la evolución y estado actual de la criptografía simétrica.

En la criptografía contemporánea existen dos tipos de algoritmos de cifrado. Estos son los cifradores de bloque y los cifradores de flujo. El cifrado en bloque consiste en dividir el

mensaje en bloques de información y después cifrarlos con la misma clave. La principal intención de estos algoritmos es difundir la información cifrada lo máximo posible a lo largo del mensaje cifrado y así eliminar la correlación de los datos originales. Los algoritmos de flujo, se basan en cifrar un único elemento de información mediante una semilla, la clave, que genera números pseudo-aleatorios. Esto quiere decir, no son números realmente aleatorios, pero siempre son de una longitud grande, de manera que sean intratables desde el punto de visto computacional. Además, consisten en una cantidad fija de cifras y otra asociada al número aleatorio anterior [9]. De esta forma se consiguen unos algoritmos que funcionan prácticamente en tiempo real.

Si comparamos un tipo de algoritmos frente al otro, vemos que el cifrador de flujo consigue mayor difusión de la información original en el mensaje cifrado y es resistente a la adición de bloques (puesto que se alteraría el mensaje). Además son algoritmos diseñados para que la acción de descifrado sea muy similar a la de cifrado, solo hay que seguir los pasos en orden inverso. Sin embargo, al tener que operar bloque a bloque resulta algo más lento que el cifrador de flujo y tiene el inconveniente de que si se produce un error en el proceso de cifrado, se propagará en todo el bloque. Los cifradores de flujos tienen la principal ventaja, como se ha mencionado antes, en la velocidad de cifrado. No propagan posibles errores en el proceso pero no distribuyen tanto la información y son más vulnerables a modificaciones que los de bloque.

Llegados a este punto podemos deducir que el cifrador que se utilizará para este proyecto se tratará de un cifrador de bloque. En la actualidad existen multitud de algoritmos clasificados como algoritmos de cifrado en bloque. Los más sencillos siguen un algoritmo basado en el que definió Horst Feistel.

El esquema Feistel consiste en partir el mensaje en claro por la mitad para operar de manera cruzada. A una de las mitades, se le aplican ciertas transformaciones (permutación de caracteres, rellenado,…) y después se aplica una función con la clave o subclave. Justo después, se realiza una operación XOR con la otra mitad y las mitades se intercambian de posición.

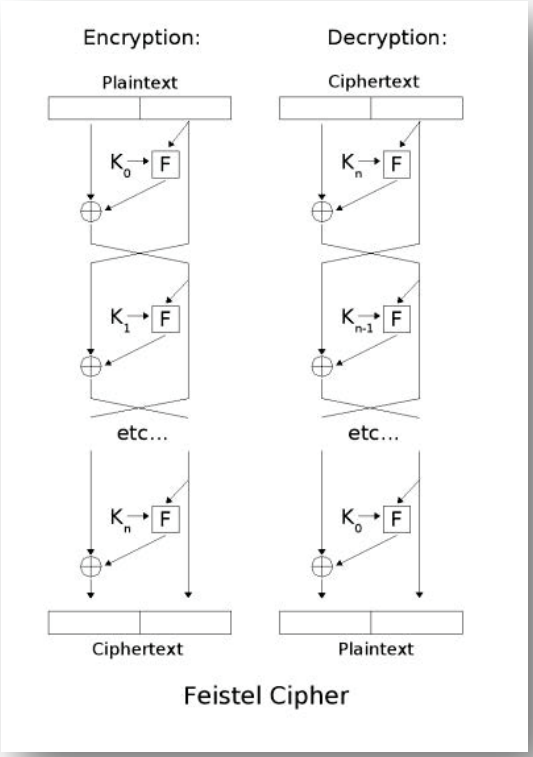


ILUSTRACIÓN 15: ESQUEMA FEISTEL

Un claro ejemplo de algoritmo de cifrado que sigue este esquema es el algoritmo DES (Data Encryption Standard). Este cifrado resultaba ser lo suficientemente fuerte y además muy veloz. No solo eso, sino que además demostraba que para realizar un descifrado, como en toda red de Feistel, bastaba con seguir los pasos de cifrado en orden inverso. La fortaleza de DES, residía en la función (que denominaremos, siguiendo el esquema función F), que constaba de expansión de bloque, mezcla de datos (con una función XOR con una de las subclaves), sustitución de datos (en combinación con matrices) y permutación de caracteres. Las subclaves (K_i en nuestro esquema) se generaban de manera sencilla, rotando un cierto número de bytes a la izquierda. Estas operaciones se realizan un número de veces, conocido como número de rondas. Además, existen diferentes modos de operación, en los que, por ejemplo, un bloque cifrado puede influir en el cifrado del siguiente, o en la generación de subclaves.

Aunque este algoritmo parecía imposible de romper en la década de 1970, cuando se diseñó, la computación ha avanzado lo suficiente como para romper este cifrado con bastante facilidad.

Por suerte, ya se estaba investigando, buscando algoritmos igual de veloces y reusables pero más fuertes. Es cuando en 2001 aparece el AES (Advanced Encryption Standard). Actualmente el gobierno de EEUU lo utiliza para el cifrado de datos de cierta importancia para la seguridad nacional. También se utiliza en transferencias bancarias.

El algoritmo AES tiene un tamaño de bloque fijo de 128 bits. El tamaño mínimo de la clave también es de 128 bits, la longitud más común es la más segura, con 256 bits. El número de rondas es variable. AES, no utiliza una red de Feistel, en su lugar, usa una red de sustitución-permutación más compleja.

Actualmente es uno de los algoritmos de cifrado más fuertes que existen. Si bien en un futuro quizá próximo sea rompible computacionalmente como ha sido el DES, se pueden (y se deben) tomar ciertas medidas adicionales para dificultar un posible ataque contra un cifrado, tanto de bloque como de flujo.

Una de las medidas más utilizadas en cifrados es la derivación de la clave, es decir, la clave no se introduce como parámetro en el algoritmo sin más. En su lugar, se le aplica cierto tratamiento, generando así lo que se conoce como subclaves. Existen infinitas formas de generar subclaves, mediante fórmulas matemáticas, o tomando como parámetro la fecha o la posición de un objeto. Lo más corriente es utilizar una cadena aleatoria conocida como *salt*, que se concatena con la subclave generada. El algoritmo de generación de subclaves suele tener más de una iteración, por evidentes motivos de seguridad.

Como medida de seguridad adicional podemos encontrar frecuentemente el uso de caracteres de relleno, sobre todo en casos de cifrados de bloque. Los caracteres de relleno, o *padding*, se utilizan sobre todo para evitar ataques de fuerza bruta contra algoritmos con contraseñas no derivadas o para paliar las debilidades de este tipo de algoritmos de repetir caracteres sistemáticamente.

1.2.4. TOPOOS

ILUSTRACIÓN 16: LOGO DE TOPOOS

Topoos es un conjunto de servicios para el desarrollo de aplicaciones sociales. De manera gratuita, proporciona funcionalidades ya implementadas para que sean incluidas en la aplicación a desarrollar. Funciona tanto para aplicaciones web como para apps móviles. Aún se encuentra en versión beta y muchas de las funcionalidades están en fase de desarrollo. No obstante los servicios que funcionan correctamente son muy útiles. Algunos de ellos son: [29].

- Identidad digital: Topoos facilita la gestión de usuarios de las aplicaciones. Desde formularios de acceso, mantenimiento y aspectos de seguridad, gracias al protocolo de autenticación OAuth 2.0. La autenticación o registro de usuarios puede ser obligatoria u opcional en una aplicación.
- Tracking en tiempo real: Un usuario de una aplicación con Topoos puede consultar su posición, encontrar otros usuarios o puntos de interés previamente definidos. La posición se registra de manera aislada o en forma de sesión posibilitando identificar sucesos importantes durante la misma.
- Puntos de interés: Capacidad para añadir, y gestionar puntos de interés. Este servicio viene de la mano del servicio de tracking. El desarrollador o incluso el usuario podrá añadir puntos de interés a una base de datos.

- Social: Topoos tiene un servicio que genera una pequeña red social en la aplicación. Permite la creación de relaciones entre usuarios, desde mensajes, seguimientos, amistad, comentarios,…
- Eventos: Se trata de un servicio que envía notificaciones a un usuario cuando se produzcan ciertos eventos, o cuando se cumplan ciertas reglas preestablecidas. Se puede combinar, por ejemplo con el servicio de tracking o con el de puntos de interés para informar a un usuario que está cerca de un punto de interés.
- Imágenes: Almacenamiento y acceso remoto a imágenes que pueden subir los usuarios.

Actualmente se están implementando servicios de alto nivel e integración con servicios de terceros. Asimismo, se están desarrollando herramientas de generación de flujos de trabajo.

Topoos se integra principalmente con el sistema operativo móvil Android, pero también existen aplicaciones web y apps para iOS. Se trata de un framework digno de analizar por las facilidades que aporta, sobre todo la gestión de usuarios, que se debe realizar en un servidor y que dificultaría mucho el desarrollo de una aplicación con este servicio.

1.2.4.1. PROTOCOLO OAUTH 2.0

El protocolo OAuth 2.0 es un protocolo abierto para la autenticación de usuarios. OAuth 2.0 es utilizado por grandes compañías como la red social Facebook, o el buscador Google. El protocolo busca reducir en la medida de lo posible el número de identificaciones (y por tanto llamadas) que debe hacer un usuario para registrarse o autenticarse en una plataforma.

Está orientado al desarrollo web con “flows”, o aplicaciones con flujo de estados. Y utiliza en gran parte el protocolo de seguridad SSL, lo que lo convierte en un protocolo seguro. Utiliza tokens de identificación, y no información comprometida como email, nombres de usuario etc., por lo que también es muy sencillo de cara al usuario final. En la siguiente imagen se muestra un sencillo ejemplo de funcionamiento.

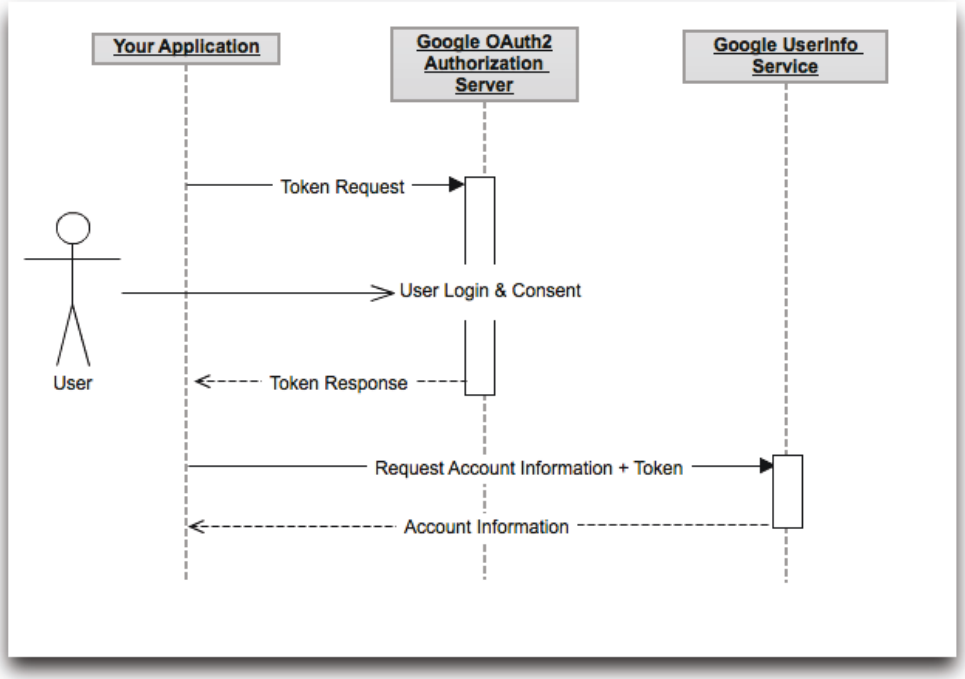


ILUSTRACIÓN 17: PROTOCOLO OAUTH 2.0

1.3. MARCO REGULADOR

Para poder desarrollar cualquier aplicación, se debe estudiar previamente las leyes vigentes y las normativas técnicas que atañen al desarrollo de la misma. Seguidamente, se procederá a definir cada una de estas leyes.

1.3.1. LEY GENERAL DE LAS TELECOMUNICACIONES

La ley general de las telecomunicaciones fundó un régimen plenamente liberalizado en la asistencia de servicios y el establecimiento y explotación de redes de telecomunicaciones, descubriendo el sector a la libre competencia entre operadores [30]. Su objetivo no es otro que el de asegurar y avalar el secreto de las comunicaciones. El artículo de esta ley al que afecta en el desarrollo del proyecto es el 36 de la Ley General de Telecomunicaciones sobre el "*Cifrado en las redes y servicios de comunicaciones electrónicas*". Este artículo pertenece a la ley datada del 3 de Noviembre de 2003 y dice así [30]:

1. "*Cualquier tipo de información que se transmita por redes de comunicaciones electrónicas podrá ser protegida mediante procedimientos de cifrado.*"

2. "*El cifrado es un instrumento de seguridad de la información. Entre sus condiciones de uso, cuando se utilice para proteger la confidencialidad de la información, se podrá imponer la obligación de facilitar a un órgano de la Administración General del Estado o a un organismo público, los algoritmos o cualquier procedimiento de cifrado utilizado, así como la obligación de facilitar sin coste alguno los aparatos de cifra a efectos de su control de acuerdo con la normativa vigente.*"

Viendo el primer punto, nos damos cuenta de que el hecho de cifrar está permitido y no se incumple dicha ley. La aplicación que se va a desarrollar va a emplear técnicas de cifrado sobre los datos que, más tarde, pueden enviarse mediante aplicaciones de terceros por redes de comunicaciones electrónicas.

El segundo punto informa de que la Administración General del Estado o un organismo público pueden requerir los algoritmos o procedimientos empleados para cifrar esa información. Se tiene intención de facilitar la aplicación de manera pública, de ese modo, cualquier organismo puede acceder a ella. Se facilitarán datos de contacto con el desarrollador, tales como el correo electrónico, de manera que se pueda contactar con él para poder reclamar la información pertinente en caso de ser necesario.

1.3.2. LEY ORGÁNICA DE PROTECCIÓN DE DATOS

La ley orgánica de protección de datos tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar. [31]

Esta Ley Orgánica, a la cual nos referiremos como LOPD, datada del 13 de Diciembre, manifiesta en el artículo 9 llamado "*Seguridad de los datos*" que:

"*El responsable del fichero, y, en su caso, el encargado del tratamiento deberán adoptar las medidas de índole técnica y organizativas necesarias que garanticen la seguridad de los datos de carácter personal y eviten su alteración, pérdida, tratamiento o acceso no autorizado, habida cuenta del estado de la tecnología, la naturaleza de los datos almacenados y los riesgos a que están expuestos, ya provengan de la acción humana o del medio físico o natural.*" [31]

Este punto señala que, tanto el usuario final de la aplicación como el desarrollador, deben tomar las medidas de seguridad pertinentes para que no se vea afectada la privacidad de los datos de carácter personal. Para respetar y cumplir con esta ley, se toman las siguientes decisiones:

1. Los algoritmos de cifrado de la información personal deben ser suficientemente seguros. De modo que ningún ataque actual contra la privacidad de los datos sea computacionalmente viable.
2. La información contenida en una imagen cifrada por la aplicación que se va a desarrollar, únicamente será accesible, por el emisor de dicha imagen y sus destinatarios, autorizados únicamente por el primero.
3. La información de la autenticación e identificación de los usuarios se realizará a través de uno de los servicios del framework Topoos. Es esta entidad la responsable del correcto almacenamiento de este tipo de información.
4. Las imágenes cifradas podrán ser enviadas en ocasiones mediante aplicaciones de terceros y medios ajenos al dispositivo móvil personal. Es responsabilidad del usuario de la aplicación el correcto uso de esta capacidad.

2. OBJETIVOS

El objetivo final del presente proyecto es el desarrollo de una aplicación que sea capaz de cubrir las necesidades, carencias e inquietudes anteriormente planteadas.

2.2. CIFRADOR DE IMÁGENES MULTICONDICIÓN

La aplicación que se va a desarrollar, se puede definir como un cifrador de imágenes multicondición para una plataforma móvil Android. Dicha aplicación tendrá un nombre con el que nos referiremos a ella de ahora en adelante. Ese nombre será "*Image Cipher*".

A diferencia de las escasas aplicaciones de este tipo que se encuentran en el mercado, Image Cipher será un cifrador multicondición, esto quiere decir, que además de la contraseña, podremos añadir ciertas condiciones que se deben cumplir para poder descifrar una imagen. Todas las condiciones necesarias se indican en el momento de cifrar y son las siguientes:

- Contraseña: Que no es obligatoria, aunque si recomendable. En la contraseña reside gran parte de la fortaleza de cifrado de la aplicación.
- Usuario: De manera que solo los usuarios autorizados que reciban la imagen sean capaces de descifrarla.
- WI-FI ID: Si se añade esta condición, la imagen únicamente será descifrable desde el punto de acceso WI-FI indicado en el momento de cifrar.
- Fecha / Hora: La foto cifrada solo será descifrable a partir de la fecha y hora indicada.

La aplicación también tendrá que ser compatible con el envío de las imágenes a través de otras aplicaciones, como por ejemplo email, sms o servicios de mensajería instantánea.

Por ultimo Image Cipher permitirá almacenar cifradas las imágenes, así como la posibilidad de eliminar la imagen después del proceso de encriptación.

3. ANÁLISIS

En primer lugar, debemos obtener una especificación detallada de lo que va a ser el sistema de información que se va a desarrollar. Dicha especificación actuará como los cimientos, la base del sistema a partir de donde se empezará a construir el diseño. El proceso de análisis trata de comprender las necesidades que tiene el usuario final de la aplicación.

3.2. ANÁLISIS SOBRE EL USO DE FRAMEWORKS

Un framework, o infraestructura digital, es un conjunto de funcionalidades y servicios que dan soporte al programador, de manera que facilita la implementación. Normalmente se tratan de pequeños artefactos o módulos software, aunque también pueden tener mayor tamaño y abarcar más capacidades.

En muchos casos, los frameworks aceleran el proceso de implementación de aplicaciones, en gran parte por la reutilización de software y la facilidad para su integración en proyectos que se encuentren en esta fase del desarrollo. Asimismo, la mayoría viene con seguridad integrada en las funcionalidades que cubren, algo que le ahorra tiempo y esfuerzo al programador.

Sin embrago, pueden resultar incomodos de incluir en un proyecto, debido a la dependencia de código que se genera en la aplicación.

La decisión que se ha tomado para el desarrollo del proyecto es la inclusión del framework analizado y descrito en apartados anteriores Topoos. La decisión se ha tomado en vista de que cubre de manera efectiva el almacenamiento de imágenes en línea y la autenticación de usuarios. Además proporciona el interesante aspecto de la geolocalización, un aspecto que, como se explicará en la definición definitiva de los objetivos (apartado “Cifrador de Imágenes Multicondición”), se desea incluir en el proyecto.

3.3. COMPRARACIÓN DE TECNOLOGÍAS

Las necesidades planteadas exigen el uso de una tecnología que el usuario final pueda llevar consigo. Es por esto por lo que se ha determinado que la aplicación a desarrollar esté orientada al menos hacia el uso de un terminal con un sistema operativo móvil.

En los apartados anteriores se han analizado los dos sistemas operativos móviles que abarcar más mercado, tanto como en dispositivos, como en aplicaciones. Se han realizado comparativas en aspectos de seguridad entre ambos sistemas, Android y iOS, llegando a las conclusiones ilustradas en la siguiente tabla [32]:

- Se ha considerado que ambos sistemas operativos tienen una madurez óptima. Tanto en Android como en iOS, se contemplan infinidad de capacidades para los terminales y se sigue investigando hoy en día para mejorarlas y añadir otras nuevas.
- En cuanto a términos de licencia, la de iOS es mucho más restrictiva que la de Android, únicamente a nivel de licencias de desarrollo, encontramos que Android es gratuito mientras que iOS impone unos precios casi prohibitivos.
- En aspectos de documentación se ha encontrado bastante más información sobre el sistema de Google, aunque la encontrada para iOS también se ha contado como información suficiente.
- Encontramos que, con la formación recibida, el desarrollador de la aplicación, encontrará más sencillo la implementación de una aplicación en Java, más que con otros lenguajes.
- Realizando un muy breve estudio de mercado, encontramos que en la actualidad existen más dispositivos con sistema Android, que con iOS.
- En cuanto al tiempo de desarrollo, se trata únicamente de una estimación, en la que para ambos sistemas se prevé un tiempo similar.
- Finalmente, vemos que la puesta a punto del entorno de programación es más sencilla para Android, por las restricciones de licencia del sistema iOS.

Por tanto, llegamos a la conclusión de que el desarrollo de la aplicación se realizará en el sistema operativo Android. Se prestará especial atención a la versión de la misma y en cierto grado a la retro-compatibilidad de la aplicación en las distintas versiones del sistema.

	Android	iOS
Madurez Tecnológica	10	10
Licencia	8	4
Herramientas de desarrollo	9	7
Documentación	10	8
Facilidad de aprendizaje	7	5
Plataformas de ejecución	9	7
Tiempos de desarrollo (estim.)	7	7
Puesta a punto del entorno	7	5
Media:	**8.375**	**6.625**

TABLA 1: COMPARACIÓN DE TECNOLOGÍAS

Se ha realizado una búsqueda por la plataforma oficial de aplicaciones de Android, Google Play. En la actualidad no existe ninguna aplicación similar a la que se quiere realizar. Sin embargo, sí que existen aplicaciones para el almacenamiento cifrado de imágenes, aunque no se contempla el uso de varias condiciones para el cifrado, ni el envío o almacenamiento en línea de las mismas.

3.4. ASPECTOS DE SEGURIDAD

Tratándose de una aplicación de almacenamiento seguro de imágenes, tenemos que prestar especial cuidado en los aspectos de seguridad. Al mismo tiempo debemos recordar que se está trabajando con dispositivos de capacidad variable y que en muchas ocasiones es muy limitada.

Aunque en un principio, se planteó sencillamente alterar unos bytes de la imagen para impedir que las aplicaciones del terminal puedan operar con ella, y eliminar esa alteración con la aplicación para que pueda ser de nuevo "legible", se optó por utilizar un cifrado que, siendo suficientemente fuerte, sea "ligero" en términos de computación, es decir, no consuma muchos recursos y sea rápido. Como se ha visto anteriormente, los cifradores simétricos de bloque son los idóneos para este trabajo.

Dados los resultados, se ha escogido el algoritmo AES, con un tamaño de clave de 256 bits y padding PKCS5.

3.5. DESARROLLO

El desarrollo del proyecto, se debe estructurar, controlar y planificar de manera previa. Para ello, en todos los desarrollos, se utilizan metodologías. Se ha considerado que para el desarrollo de Image Cipher, la metodología idónea es la Metodología de Cascada, cuyo nombre viene dado por el hecho de que sus etapas sigan un orden descendente. Las etapas se siguen de manera secuencial y se ilustran en el siguiente esquema:

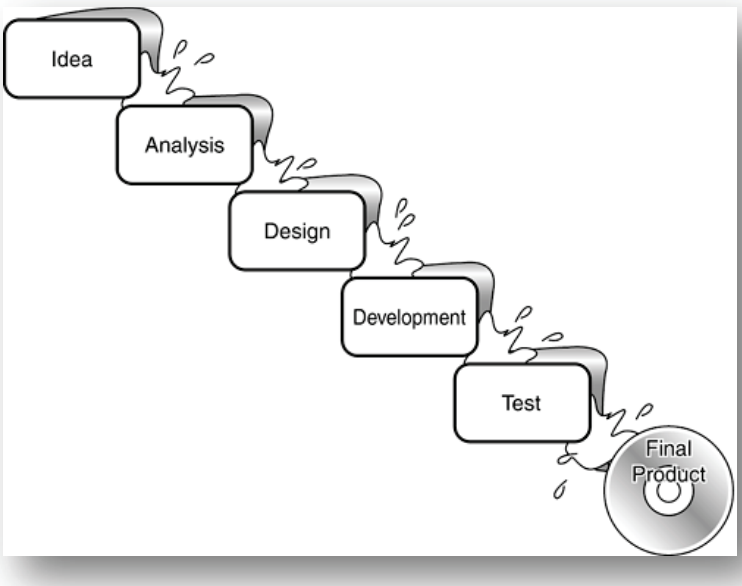


ILUSTRACIÓN 18: METODOLOGÍA EN CASCADA

Vemos pues que los pasos a seguir son: Análisis de los requisitos de la aplicación, Diseño de la solución, Implementación de la aplicación en sí y Realización de pruebas. De esta manera obtendremos el producto final, que no es otro que el propio Image Cipher.

3.6. DEFINICIÓN DEL SISTEMA

Antes de empezar a captar los requisitos de la aplicación, debemos especificar qué problemas debe solucionar el sistema, determinar sus restricciones si las hubiera y definir el entorno operacional requerido para el correcto funcionamiento de la solución.

3.6.1. ALCANCE DEL SISTEMA

El sistema de información que se va a desarrollar consiste en una app móvil que permite al usuario de un smartphone con sistema operativo Android cifrar una determinada imagen con una o varias condiciones. Asimismo, permitirá descifrar las imágenes que se generen con la aplicación en el caso de que se cumplan las condiciones necesarias para tal acción. Además, permitirá al usuario enviar las imágenes con aplicaciones de terceros, o por cualquier otro medio disponible por el usuario en su terminal.

La aplicación, no solo debe permitir cifrar imágenes, debe hacerlo de forma efectiva y rápida. Además la aplicación debe ser capaz de descifrar otras imágenes recibidas si han sido cifradas con Image Cipher y además cumplan las condiciones necesarias.

Para cifrar una imagen con Image Cipher, se permite al usuario añadir las condiciones de descifrado especificadas en el apartado "Cifrador de Imágenes multicondición" del presente documento. A la hora de descifrar, la especificación de estas condiciones permanece invisible para el usuario (no se requiere al usuario especificar qué condiciones debe cumplir la imagen para descifrarla). En caso de no cumplir una de ellas, si se informa del dicho incumplimiento.

3.6.2. RESTRICCIONES GENERALES

Las primeras restricciones que se deben cumplir son las especificadas en el apartado "Marco Regulador" de este documento. Se trata de los aspectos legales que se cumplen en el desarrollo de este proyecto y que han de ser tenidos muy en cuenta. Además deberán cumplir las siguientes restricciones:

- La interfaz de la aplicación debe ser usable, accesible y amigable.
- El lenguaje de programación para el desarrollo de la aplicación es Java.
- La aplicación debe ser compatible con el mayor número de versiones del sistema operativo móvil Android.
- La aplicación cumplirá de manera garantizada todas las funcionalidades especificadas en los requisitos de usuario.
- Los requisitos de usuario se corresponden con las decisiones acordadas entre el desarrollador y el tutor, complementados con las decisiones propias por parte del alumno.

3.6.3. ENTORNO OPERACIONAL

La aplicación que se pretende desarrollar tiene una serie de requerimientos mínimos para su ejecución. Cumpliendo estos requerimientos mínimos, la aplicación funcionará de manera satisfactoria. Dichos requerimientos serán los siguientes:

- Un dispositivo móvil con sistema operativo móvil Android con versión igual o superior a Android 2.3.0 Gingerbread. No obstante se recomienda el uso de la versión 3.0 Honeycomb, debido a que es a partir de esta versión cuando se comienza a incluir una API con un estándar de diseño denominado "Holo" que utilizará Image Cipher.
- El terminal además, deberá tener una memoria RAM no menor a 400MB. Aun así, de nuevo se recomienda el uso de al menos 700MB, para tratar con rapidez suficiente los datos y las herramientas de la aplicación.
- El dispositivo deberá tener espacio en memoria suficiente como para poder instalar la aplicación, y tener imágenes disponibles.

3.7. ENTORNO DE DESARROLLO

En este apartado se describen los elementos tanto software como hardware que componen el entorno de desarrollo con el que se creará la aplicación.

3.7.1. EQUIPOS

Se usarán los siguientes dispositivos físicos para el desarrollo de la aplicación:

- Dispositivo móvil Samsung Galaxy Ace:
  - Modelo: GT-S5830
  - Sistema Operativo: Android 2.3.7 Gingerbread
  - Procesador: Qualcomm MSM7227 800 MHz
  - RAM: 480 MB
- Dispositivo móvil Samsung Galaxy S III- Mini:
  - Modelo:
  - Sistema Operativo: Android 4.1 Jelly Bean
  - Procesador: NovaThor 1GHz
  - RAM: 1GB
- Computador Portátil LG:
  - Modelo: A-505
  - Sistema Operativo: Ubuntu 12.4
  - Procesador: Intel Core i5-
  - RAM: 4GB DDR3
  - Memoria en disco: 500GB

3.7.2. LENGUAJES DE PROGRAMACIÓN

Los lenguajes de programación utilizados han sido los siguientes:

- UNIX: Comandos básicos para interactuar con el sistema operativo del portátil y realizar la instalación del entorno.
- Java 7: Para la mayoría de la implementación de la aplicación y la integración de las librerías.

- Android 9 y 11: Aunque muy similar a Java tiene sus propias extensiones y conceptos.

3.7.3. ENTORNO DE DESARROLLO

En lo referente a elementos de software utilizados para el desarrollo del proyecto, se ha prestado especial atención a la compatibilidad con el sistema operativo del portátil. Por suerte, no se han encontrado pegas a la hora de la puesta a punto del entorno, dado que se trata de un entorno libre y gratuito, además de estar disponible para cualquier sistema operativo "de sobremesa".

Para la implementación de Image Cipher se ha utilizado el entorno Android SDK (Software Development Kit) basado en el conocido Eclipse. Android SDK, al igual que Eclipse, está programado en Java, lenguaje en el cual fija sus capacidades para el desarrollo de aplicaciones. La diferencia entre ambos entornos es simple. Android SDK incluye una serie de herramientas que no están disponibles para la versión tradicional de Eclipse. Estas herramientas son las que permiten el desarrollo de apps para el sistema operativo móvil.

De dichas herramientas, la más útil sin duda es el emulador de Android, que permite la simulación de un terminal móvil en el equipo con el SDK instalado. No solo permite la ejecución de la aplicación para comprobar el funcionamiento, sino que también permite la depuración de la app, con las facilidades que eso conlleva.

Para la depuración se ha utilizado la herramienta Logcat, perteneciente también al entorno Android. Logcat permite, el uso de logs, pero también el etiquetado de entradas y el color de las mismas para identificar de manera rápida errores, avisos, información, etc.

No se ha utilizado únicamente un entorno para la programación de Image Cipher. En cuestiones de análisis y diseño se ha usado una aplicación perteneciente al navegador Google Chrome, Gliffy, enfocada a la creación de diagramas. Esta herramienta es gratuita y libre. La única restricción que presenta es que funciona exclusivamente si se dispone de dicho navegador. Al funcionar sobre el navegador, puede usarse independientemente del sistema operativo del equipo.

3.8. REQUISITOS DE USUARIO

La captación de requisitos de usuario en un caso de proyecto de software real, se obtendría de reuniones con el cliente. En esas reuniones, el cliente expondría sus inquietudes y lo que quiere conseguir con el desarrollo. El o los analistas obtendrían una lista de requisitos y determinarían que es lo que el cliente necesita. Sin embargo, en el desarrollo del presente proyecto no hay cliente, y los requisitos de usuario obtenidos son el resultado de las reuniones realizadas entre el alumno y el tutor.

Estas reuniones tienen como fin enumerar los procesos del sistema de información, que no será otro que la app Image Cipher. Dada la naturaleza del producto, se dividirán los requisitos de usuario en dos grupos. Los requisitos de capacidad, que describirán lo que el usuario será capaz de hacer con la aplicación y los requisitos de restricción, que indicarán lo que el usuario final del producto no podrá hacer con él.

La definición e identificación de requisitos es algo compleja, por ello se realizará en forma de tablas con los siguientes atributos:

- Identificador: Es necesario para nombrar el requisito de forma unívoca. De esta manera se puede realizar un seguimiento y una identificación casi inmediata. Cada identificador está formado por varias componentes:
    - Siglas: Determinarán el tipo de requisito al cual nos estamos refiriendo, capacidad o restricción. Los requisitos de capacidad llevarán las siglas RUC (Requisito de Usuario y Capacidad), mientras que los de restricción llevaran las siglas RUR (Requisito de Usuario y Restricción).
    - Número: La numeración del requisito, puesto que no se estima que el número total de ningún tipo de requisitos de usuario sea excesivamente elevado, el número siempre contará con dos cifras.

    Vemos por tanto que un requisito con el identificador RUR-03 se corresponde con el requisito de usuario de restricción número tres.

- Nombre: Nombre del requisito. Es importante que el nombre sea relativamente descriptivo y no ha de ser unívoco.
- Descripción: En este campo se detalla en una extensión moderada lo que el requisito permite, en el caso de estar identificado como capacidad, o no permite hacer, en el caso de restricción. También debe definir en qué consiste el requisito descrito.
- Prioridad: Define la prioridad de un requisito frente a los demás. Los requisitos de mayor prioridad tendrán mayor importancia en la fase de diseño e implementación que los de prioridad menor. Se ha optado por definir la prioridad con tres niveles y de manera no numérica, por la confusión que conlleva. Los niveles de prioridad serán Alta, Media y Baja.
- Necesidad: Indica la importancia de incluir el requisito en las fases posteriores de diseño e implementación. Toma los valores Necesario y Opcional.
    - Necesario: El requisito debe ser obligatoriamente incluido en el diseño y la implementación.
    - Opcional: Dada la baja importancia del requisito, no es obligatorio su diseño ni su implementación.
- Estabilidad: Determina lo resistente que es un requisito de usuario frente a posibles cambios que se puedan producir en el sistema. Este atributo toma los valores Alta y Baja.
    - Alta: No está previsto que el requisito se altere durante el proceso de desarrollo
    - Baja: El requisito puede ser alterado en fases futuras.

La siguiente tabla muestra un ejemplo de un requisito de usuario:

RUX-YY			
Nombre:	Nombre del requisito		
Descripción:	Descripción, detalles y especificaciones del requisito		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 2: EJEMPLO DE REQUISITO DE USUARIO

3.8.1. REQUISITOS DE CAPACIDAD

Los requisitos de usuario de capacidad son aquellos que definen las funciones que se deben cubrir para resolver un problema. Describen lo que el usuario podrá hacer con el producto final. Los requisitos de capacidad obtenidos son los siguientes:

RUC-01			
Nombre:	Registro de usuario (sign up)		
Descripción:	El usuario podrá registrarse en la aplicación.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 3: RUC-01

RUC-02			
Nombre:	Autenticación de usuario (log in).		
Descripción:	El usuario podrá identificarse automáticamente sin necesidad de introducir contraseñas.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 4: RUC-02

RUC-03			
Nombre:	Error en autenticación.		
Descripción:	En caso de error en la autenticación, se informará al usuario y éste deberá autenticarse de manera manual		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 5: RUC-03

RUC-04			
Nombre:	Modo cifrado.		
Descripción:	El usuario podrá entrar en el modo de cifrado de imágenes desde la pantalla principal de la aplicación.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 6: RUC-04

RUC-05			
Nombre:	Modo descifrado.		
Descripción:	El usuario podrá entrar en el modo de descifrado de imágenes desde la pantalla principal de la aplicación.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 7: RUC-05

RUC-06			
Nombre:	Abrir imagen.		
Descripción:	El usuario podrá abrir una imagen para cifrarla o descifrarla.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 8: RUC-06

RUC-07			
Nombre:	Cifrar con contraseña.		
Descripción:	El usuario podrá introducir una contraseña para cifrar una imagen.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 9: RUC-07

RUC-08			
Nombre:	Cifrar con WI-FI ID actual.		
Descripción:	El usuario podrá establecer como condición para el descifrado de la imagen que se está cifrando, el que únicamente pueda descifrarse si el terminal que pretende descifrar esté conectado al punto de acceso WI-FI al que el usuario está conectado.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 10: RUC-08

RUC-09			
Nombre:	Cifrar con WI-FI ID introducido.		
Descripción:	El usuario podrá establecer como condición para el descifrado de la imagen que se está cifrando, el que únicamente pueda descifrarse si el terminal que pretende descifrar esté conectado al punto de acceso WI-FI que el usuario introduzca.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 11: RUC-09

RUC-10			
Nombre:	Cifrar con fecha y hora.		
Descripción:	El usuario podrá establecer como condición para el descifrado de la imagen que se está cifrando, el que únicamente pueda descifrarse si la fecha y hora establecida es menor que la fecha en la que se intente descifrar la imagen.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 12: RUC-10

RUC-11			
Nombre:	Cifrar con usuario.		
Descripción:	El usuario podrá establecer como condición para el descifrado de la imagen que se está cifrando, el que únicamente pueda descifrarse si un usuario que pretende descifrarla se encuentra entre los usuarios autorizados por el usuario que cifra.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 13: RUC-11

RUC-12			
Nombre:	Escoger modo de envío.		
Descripción:	El usuario podrá elegir de qué modo prefiere enviar una imagen cifrada.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☐Necesario	☑Opcional	
Estabilidad:	☐Alta	☑Baja	

TABLA 14: RUC-12

RUC-13			
Nombre:	Almacenamiento de una imagen cifrada.		
Descripción:	El usuario podrá elegir almacenar la imagen cifrada en la memoria del terminal.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 15: RUC-13

RUC-14			
Nombre:	Borrado de originales.		
Descripción:	El usuario podrá, en el momento de cifrar y como opción secundaria, eliminar la imagen original después de su cifrado.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☐Necesario	☑Opcional	
Estabilidad:	☐Alta	☑Baja	

TABLA 16: RUC-14

RUC-15			
Nombre:	Descifrado de una imagen.		
Descripción:	Un usuario podrá descifrar una imagen si cumple todas las condiciones establecidas en el momento de su cifrado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 17: RUC-15

RUC-16			
Nombre:	Almacenamiento de una imagen descifrada.		
Descripción:	Se debe informar al usuario que la imagen descifrada se ha almacenado en la memoria del terminal, así como de la ruta de almacenamiento.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 18. RUC-16

RUC-17			
Nombre:	Navegación sencilla.		
Descripción:	La navegación a través de la aplicación debe ser sencilla e intuitiva. Se informará al usuario sobre el uso de cada acción que puede realizar.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 19: RUC-17

RUC-18			
Nombre:	Botón atrás.		
Descripción:	El usuario podrá volver un paso atrás mediante el uso del botón "Back" de Android.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 20: RUC-18

3.8.2. REQUISITOS DE RESTRICCIÓN

Los requisitos de usuario de restricción definen cualquier tipo de limitación en la que se llevan a cabo las funcionalidades de la aplicación. Estos requisitos también detallan como se llega a la solución del problema planteado. Los requisitos de restricción obtenidos son los siguientes:

RUR-01			
Nombre:	Log in incorrecto.		
Descripción:	La aplicación no debe permitir la autenticación de un usuario que haya introducido mal su contraseña o su nombre de usuario.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 21: RUR-01

RUR-02			
Nombre:	Tamaño de la imagen.		
Descripción:	La aplicación debe controlar el tamaño de las imágenes con las que trata. El tamaño de la imagen debe estar comprendido entre 15KB (512x512px) por motivos de seguridad y 4MB por motivos de rendimiento.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☐Alta	☑Baja	

TABLA 22: RUR-02

RUR-03			
Nombre:	Privacidad de la contraseña al cifrar.		
Descripción:	La aplicación debe ocultar la contraseña que se está introduciendo al cifrar con asteriscos.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 23: RUR-03

RUR-04			
Nombre:	Privacidad de la contraseña al descifrar.		
Descripción:	La aplicación debe ocultar la contraseña de descifrado con asteriscos.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 24: RUR-04

RUR-05			
Nombre:	Seguridad de la contraseña.		
Descripción:	La aplicación debe controlar que las contraseñas introducidas, tanto como para cifrar como para descifrar, sean suficientemente seguras.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 25: RUR-05

RUR-06			
Nombre:	Optimización de seguridad.		
Descripción:	La aplicación debe tomar medidas para que los algoritmos criptográficos sean consistentes.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 26: RUR-06

RUR-07			
Nombre:	Dependencia de acceso internet.		
Descripción:	La aplicación requerirá conexión a internet por alguno de los medios disponibles en el dispositivo (WI-FI, 3G,…) para la autenticación y el registro del usuario, así como para la comprobación de la fecha en el momento de descifrar.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 27: RUR-07

RUR-08			
Nombre:	Tiempos de cifrado.		
Descripción:	El tiempo de cifrado o descifrado, en el peor de los casos, no excederá de 1 minuto.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☐Necesario	☑Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 28: RUR-08

RUR-09			
Nombre:	No descifrar si no se cumplen condiciones.		
Descripción:	La aplicación en ningún caso deberá descifrar una imagen si un usuario no reúne las condiciones necesarias para ello.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	

TABLA 29: RUR-09

RUR-10			
Nombre:	Informar de condiciones no cumplidas.		
Descripción:	Si al descifrar una imagen, un usuario no reúne las condiciones necesarias para ello, la aplicación le informará con un mensaje.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☐Alta	☑Baja	

TABLA 30: RUR-10

3.9. CASOS DE USO

En este apartado se describirán los casos de uso del sistema de información. Los casos de uso son fruto de los requisitos detallados hasta este punto. El fin de los casos de uso no es otro que describir las capacidades del usuario sobre el sistema. Un caso de uso puede satisfacer uno o más requisitos de usuario.

Para definir los casos de uso, en primer lugar, se ilustrarán en un diagrama. Posteriormente, se realizarán varios diagramas más específicos detallando todos los casos de uso. Finalmente se añadirá una serie de tablas con todos los atributos de cada caso de uso. Dichos atributos se detallan a continuación:

- Identificador: De nuevo, requeriremos un identificador univoco para referirnos a cada uno de los casos de uso. El identificador estará formado por las siglas CU (Caso de Uso) y una numeración incremental de dos cifras. Por ejemplo el caso de uso con identificador CU-02 equivaldría al segundo de los casos de uso.
- Nombre: Se trata de un nombre descriptivo del caso de uso.
- Actores: Define que actor, que en este caso se tratará de usuarios de la aplicación, interactuará con el caso de uso.
- Descripción: Define de qué manera el actor interactúa con el sistema y la respuesta que recibe por par de éste.
- Condiciones previas: Define el estado del sistema que necesita el caso de uso para cumplirse.
- Post-condiciones: Detalla el estado del sistema que surge como resultado de la aplicación del caso de uso.
- Secuencia: Especifica el orden de los pasos necesarios para realizar el caso de uso.
- Secuencia alternativa: Especifica otro orden de pasos, si lo hubiera.
- Requisitos: Enumera los requisitos de usuario relacionados con el caso de uso.

CU-XX	
Nombre:	Nombre del caso de uso
Actores:	Actores que intervienen
Descripción:	Descripción detallada del caso de uso
Condiciones Previas:	Condiciones necesarias
Postcondiciones:	Condiciones resultantes
Secuencia:	Secuencia principal de ejecución
Secuencia Alternativa:	Secuencia alternativa
Requisitos:	Requisitos involucrados

TABLA 31: EJEMPLO DE CASO DE USO

En la siguiente ilustración se detalla el modelo de casos de uso:

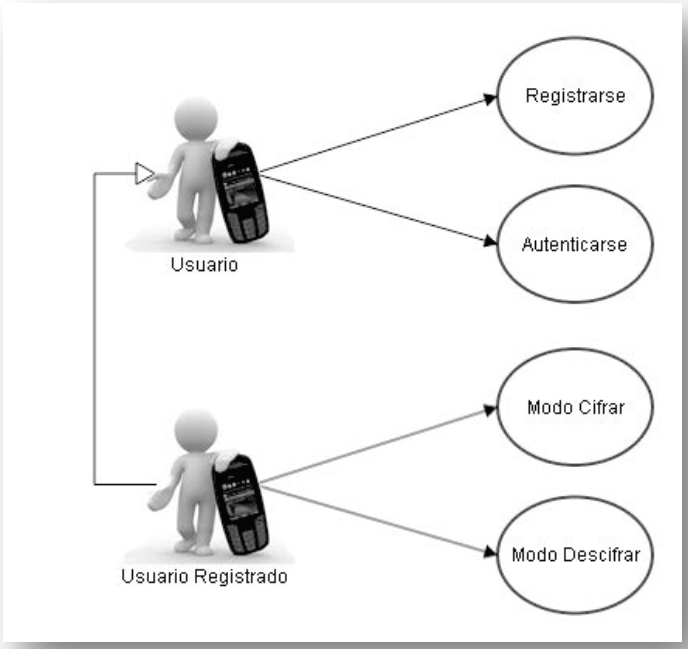


ILUSTRACIÓN 19: CASOS DE USO (GENERAL)

Tal y como se ve en el esquema, existen más de un tipo de actor, que realizará uno u otro caso de uso. Un usuario general podrá acceder a los casos de uso de registrarse o de autenticarse. El cumplimiento satisfactorio de ambos casos de uso provoca que el usuario general se transforme en un usuario registrado. Este tipo de actor extiende del usuario general. Basándose en qué caos de uso seleccione el usuario registrado, pasará a poder cumplir los casos de uso del modo cifrar o el modo descifrar, que se describirán más adelante.

Los casos de uso generales son los siguientes:

CU-01	
Nombre:	Registrarse.
Actores:	Usuario.
Descripción:	Un usuario de la aplicación que no dispone de registro previo pretende darse de alta en la aplicación.
Condiciones Previas:	Ejecutar la aplicación, conexión a internet.
Postcondiciones:	El usuario se registra y pasa a ser un usuario registrado.
Secuencia:	1. Ejecutar la aplicación. 2. Rellenar el formulario. 3. Validar formulario.
Secuencia Alternativa:	
Requisitos:	RUC-01, RUC-17, RUR-05, RUR-07.

TABLA 32: CU-01

CU-02	
Nombre:	Autenticarse
Actores:	Usuario
Descripción:	Un usuario que dispone de registro previo pretende acceder a la aplicación.
Condiciones Previas:	Ejecutar la aplicación, conexión a internet, haberse registrado con anterioridad
Postcondiciones:	El usuario podrá acceder a los modos de cifrado o descifrado.
Secuencia:	Se realiza automáticamente mediante autenticación por token.
Secuencia Alternativa:	1. Ejecutar la aplicación. 2. Rellenar el formulario. 3. Validar formulario.
Requisitos:	RUC-02, RUC-03, RUC-17, RUR-01, RUR-05, RUR-07.

TABLA 33: CU-02

CU-03	
Nombre:	Modo Cifrar.
Actores:	Usuario Registrado.
Descripción:	Un usuario registrado podrá acceder al modo cifrar.
Condiciones Previas:	Ejecutar la aplicación.
Postcondiciones:	El usuario podrá realizar los casos de uso del modo cifrar.
Secuencia:	1. Ejecutar la aplicación. 2. Seleccionar “Cifrar Imagen”.
Secuencia Alternativa:	
Requisitos:	RUC-04, RUC-17, RUR-06, RUR-08.

TABLA 34: CU-03

CU-04	
Nombre:	Modo Descifrar.
Actores:	Usuario Registrado.
Descripción:	Un usuario registrado podrá acceder al modo descifrar.
Condiciones Previas:	Ejecutar la aplicación.
Postcondiciones:	El usuario podrá realizar los casos de uso del modo descifrar.
Secuencia:	1. Ejecutar la aplicación. 2. Seleccionar “Descifrar Imagen”.
Secuencia Alternativa:	
Requisitos:	RUC-05, RUC-17, RUR-06, RUR-08.

TABLA 35: CU-04

3.9.1. CASOS DE USO DEL MODO CIFRAR

En este apartado se describen los casos de uso que un usuario puede realizar en el modo de cifrado de la aplicación. A continuación se muestra un esquema, más completo, en el que se muestran dichos casos de uso.

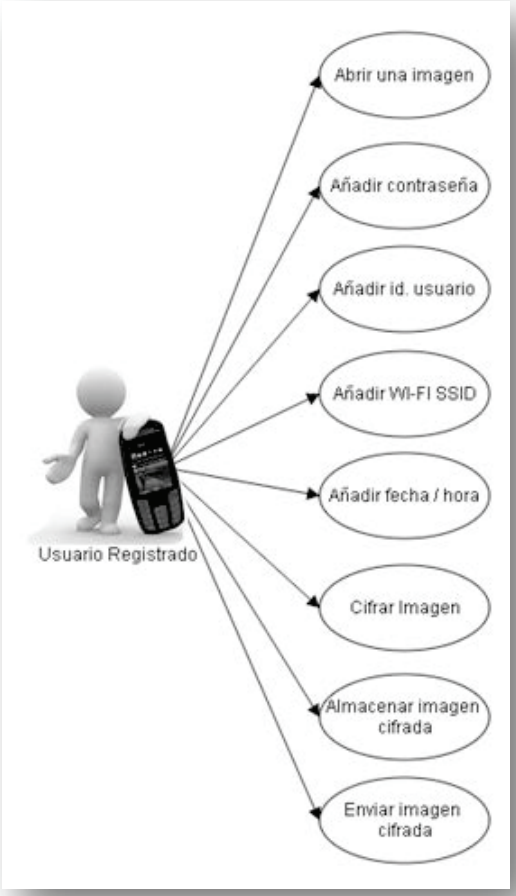


ILUSTRACIÓN 20: CASOS DE USO (CIFRAR)

Vemos que para cifrar una imagen, el actor, obligatoriamente un usuario registrado, tiene varias posibilidades a la hora de elegir las condiciones de descifrado. Cada una de estas condiciones son opcionales y el usuario puede elegir no añadirlas.

Las tablas siguientes describen los casos de uso propios de este modo.

CU-05	
Nombre:	Abrir una imagen.
Actores:	Usuario registrado.
Descripción:	El usuario selecciona una imagen para cifrar.
Condiciones Previas:	Ejecutar la aplicación, modo cifrar.
Postcondiciones:	Se carga la imagen que se va a cifrar.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar “Cifrar imagen”. 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección.
Secuencia Alternativa:	
Requisitos:	RUC-04, RUC-06, RUC-17, RUC-18, RUR-02.

TABLA 36: CU-05

CU-06	
Nombre:	Añadir contraseña.
Actores:	Usuario registrado.
Descripción:	El usuario introduce una contraseña para el cifrado de la imagen seleccionada.
Condiciones Previas:	Ejecutar la aplicación, modo cifrar, haber seleccionado una imagen.
Postcondiciones:	Se carga la contraseña para cifrar la imagen.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Añadir Contraseña". 6. Introducir contraseña deseada. 7. Confirmar.
Secuencia Alternativa:	
Requisitos:	RUC-04, RUC-07, RUC-17, RUC-18, RUR-03, RUR-05.

TABLA 37: CU-06

CU-07	
Nombre:	Añadir id. Usuario.
Actores:	Usuario registrado.
Descripción:	El usuario introduce el identificador de usuario de Topoos del usuario destinatario de la imagen que se está cifrando.
Condiciones Previas:	Ejecutar la aplicación, modo cifrar, haber seleccionado una imagen.
Postcondiciones:	Se carga el identificador para cifrar la imagen.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Añadir Usuario". 6. Introducir el usuario deseado. 7. Confirmar.
Secuencia Alternativa:	
Requisitos:	RUC-04, RUC-11, RUC-17, RUC-18.

TABLA 38: CU-07

CU-08	
Nombre:	Añadir WI-FI SSID.
Actores:	Usuario registrado.
Descripción:	El usuario selecciona el WI-FI SSID del punto de acceso a internet del destinatario.
Condiciones Previas:	Ejecutar la aplicación, modo cifrar, haber seleccionado una imagen.
Postcondiciones:	Se carga el WI-FI SSID para cifrar la imagen.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Añadir WI-FI SSID". 6. Introducir WI-FI SSID deseado. 7. Confirmar.
Secuencia Alternativa:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Añadir WI-FI SSID". 6. Seleccionar "Utilizar Actual". 7. Confirmar.
Requisitos:	RUC-04, RUC-08, RUC-09, RUC-17, RUC-18, RUR-07.

TABLA 39: CU-08

CU-09	
Nombre:	Añadir fecha / hora.
Actores:	Usuario registrado.
Descripción:	El usuario selecciona la fecha y la hora en la que la imagen podrá ser descifrada.
Condiciones Previas:	Ejecutar la aplicación, modo cifrar, haber seleccionado una imagen.
Postcondiciones:	Se carga la fecha y la hora para cifrar la imagen.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Añadir Fecha / Hora". 6. Seleccionar fecha en el formulario. 7. Seleccionar hora en el formulario. 8. Confirmar.
Secuencia Alternativa:	
Requisitos:	RUC-04, RUC-10, RUC-17, RUC-18.

TABLA 40: CU-09

CU-10	
Nombre:	Cifrar imagen.
Actores:	Usuario registrado.
Descripción:	Se reúnen todas las condiciones de cifrado, junto con la imagen para cifrarla.
Condiciones Previas:	Ejecutar la aplicación, modo cifrar, haber seleccionado una imagen, haber terminado de seleccionar las condiciones (al ser opcionales, no se necesitan todas).
Postcondiciones:	La imagen se cifra con las condiciones seleccionadas.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Cifrar".
Secuencia Alternativa:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Eliminar original". 6. Seleccionar "Cifrar".
Requisitos:	RUC-04, RUC-14, RUC-17, RUR-06, RUR-08.

TABLA 41: CU-10

CU-11	
Nombre:	Almacenar imagen cifrada.
Actores:	Usuario registrado.
Descripción:	Una vez cifrada, el usuario podrá elegir almacenar la imagen en el terminal.
Condiciones Previas:	Ejecutar aplicación, modo cifrar, haber cifrado una imagen.
Postcondiciones:	Se almacena la imagen en la memoria del teléfono.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Cifrar". 6. Seleccionar "Almacenar imagen".
Secuencia Alternativa:	
Requisitos:	RUC-04, RUC-13, RUC-17, RUC-18.

TABLA 42: CU-11

CU-12	
Nombre:	Enviar imagen cifrada.
Actores:	Usuario registrado.
Descripción:	Una vez cifrada, el usuario podrá elegir enviar la imagen al destinatario.
Condiciones Previas:	Ejecutar aplicación, modo cifrar, haber cifrado una imagen.
Postcondiciones:	Se envía la imagen al destinatario.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Cifrar imagen". 3. Seleccionar la imagen que se quiere cifrar. 4. Confirmar selección. 5. Seleccionar "Cifrar". 6. Seleccionar "Almacenar imagen".
Secuencia Alternativa:	
Requisitos:	RUC-04, RUC-12, RUC-17, RUC-18, RUR-17.

TABLA 43: CU-12

3.9.2. CASOS DE USO DEL MODO DESCIFRAR

En este apartado se describen los casos de uso que un usuario puede realizar en el modo de descifrado de imágenes de la aplicación. A continuación se muestra un esquema, más completo, en el que se muestran dichos casos de uso.

ILUSTRACIÓN 21: CASOS DE USO (DESCIFRAR)

En este caso el actor vuelve a ser el usuario registrado y que, para este modo, no es el usuario el que elige como descifrar la imagen, sino la propia aplicación. Éste se encargará de detectar automáticamente el cumplimiento de las condiciones de descifrado, es decir, se realizarán las comprobaciones de manera transparente al usuario.

Las tablas siguientes describen los casos de uso propios de este modo.

CU-13	
Nombre:	Abrir imagen cifrada.
Actores:	Usuario registrado.
Descripción:	El usuario abre una imagen con el fin de descifrarla para ver su contenido.
Condiciones Previas:	Ejecutar aplicación, modo descifrar, imagen cifrada válida.
Postcondiciones:	Se carga la imagen para descifrarla.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Descifrar Imagen". 3. Seleccionar la imagen a descifrar. 4. Confirmar selección.
Secuencia Alternativa:	
Requisitos:	RUC-05, RUC-06, RUC-17, RUC-18, RUR-02.

TABLA 44: CU-13

CU-14	
Nombre:	Descifrar imagen.
Actores:	Usuario registrado.
Descripción:	La aplicación descifra la imagen cifrada.
Condiciones Previas:	Ejecutar la aplicación, modo descifrar, haber seleccionado previamente una imagen cifrada válida.
Postcondiciones:	La imagen se descifra pudiendo el usuario ver su contenido.
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Descifrar Imagen". 3. Seleccionar la imagen a descifrar. 4. Confirmar selección. 5. Seleccionar "Descifrar".
Secuencia Alternativa:	
Requisitos:	RUC-05, RUC-15, RUC-17, RUC-18, RUR-04, RUR-06, RUR-07, RUR-08, RUR-09, RUR-10.

TABLA 45: CU-14

CU-15	
Nombre:	Almacenar imagen descifrada.
Actores:	Usuario registrado.
Descripción:	La aplicación guardará la imagen descifrada.
Condiciones Previas:	Ejecutar aplicación, modo descifrar, tener cargada una imagen descifrada.
Postcondiciones:	La imagen descifrada se almacena en la memoria del terminal
Secuencia:	1. Ejecutar aplicación. 2. Seleccionar "Descifrar Imagen". 3. Seleccionar la imagen a descifrar. 4. Confirmar selección. 5. Seleccionar "Descifrar". 6. Seleccionar "Guardar Imagen"
Secuencia Alternativa:	
Requisitos:	RUC-05, RUC-16, RUC-17.

TABLA 46: CU-15

3.10. REQUISITOS DE SOFTWARE

Los requisitos d software son el producto que se obtiene a partir de los requisitos de usuario y los casos de uso ya descritos. Estos requisitos, definen las funcionalidades que tendrá la aplicación, en otras palabras, lo que el sistema debe hacer.

Puesto que los requisitos son muy numerosos, podemos agruparlos en distintos tipos, tales como los siguientes:

- Requisitos de software funcionales: Definen lo que la aplicación tiene que hacer. Son obtenidos a partir de los casos de uso.
- Requisitos de software no funcionales: Definen como deben realizarse los requisitos funcionales del sistema de información. Se dividen en varios sub-tipos:
    - Requisitos de operación: Definen como debe realizar las tareas el sistema de información.
    - Requisitos de interfaz: Definen como será la manera que tiene el usuario de interactuar con el sistema y sus distintos módulos.
    - Requisitos de rendimiento: Describen la carga computacional, temporal, de almacenamiento, etc. que tendrá el sistema.
    - Requisitos de recursos: Indican los recursos que son necesarios para que el sistema funcione correctamente.
    - Requisitos de Seguridad: Especifican que aspectos de seguridad debe cumplir el sistema.
    - Requisitos de verificación: Indican las limitaciones que sufre el sistema al verificar los datos de entrada y salida.

Puesto que la estructuración y enumeración de los requisitos de software es más compleja aún que la de los requisitos de usuario, se volverá a utilizar un formato tabular para su definición. Cada tabla, pues, contará con los siguientes atributos:

- Identificador: Vuelve a ser necesario para nombrar el requisito de forma unívoca. De esta manera se puede realizar un seguimiento y una rápida identificación. Cada identificador está formado por varias componentes:
    - Siglas: Determinarán el tipo de requisito al cual nos estamos refiriendo, capacidad o restricción. Los requisitos de capacidad llevarán las siglas RS (Requisito de Software).
    - Número: La numeración del requisito, puesto que no se estima que el número total de ningún tipo de requisitos de usuario sea excesivamente elevado, el número siempre contará con dos cifras.

    Vemos por tanto que un requisito con el identificador RS-05 se corresponde con el requisito de usuario de restricción número cinco.

- Nombre: Nombre del requisito. Es importante que el nombre sea relativamente descriptivo y no ha de ser unívoco.
- Descripción: En este campo se detalla en una extensión moderada en qué consiste el requisito.
- Prioridad: Define la prioridad de un requisito frente a los demás. Los requisitos de mayor prioridad tendrán mayor importancia en la fase de diseño e implementación que los de prioridad menor. Se ha optado por definir la prioridad con tres niveles y de manera no numérica, por la confusión que conlleva. Los niveles de prioridad serán Alta, Media y Baja.

- Necesidad: Indica la importancia de incluir el requisito en las fases posteriores de diseño e implementación. Toma los valores Necesario y Opcional.
    - Necesario: El requisito debe ser obligatoriamente incluido en el diseño y la implementación.
    - Opcional: Dada la baja importancia del requisito, no es obligatorio su diseño ni su implementación.
- Estabilidad: Determina lo resistente que es un requisito de software frente a posibles cambios que se puedan producir en el sistema. Este atributo toma los valores Alta y Baja.
    - Alta: No está previsto que el requisito se altere durante el proceso de desarrollo
    - Baja: El requisito puede ser alterado en fases futuras.

La siguiente tabla muestra un ejemplo de un requisito de software:

RS-XX			
Descripción:			
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:			

TABLA 47: EJEMPLO DE REQUISITO DE SOFTWARE

3.10.1. REQUISITOS FUNCIONALES

Los requisitos de software funcionales, definen lo que la aplicación debe hacer.

RS-01			
Descripción:	El sistema de información deberá permitir al usuario registrarse en la aplicación.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-01, RUC-03, RUR-01, RUR-07		

TABLA 48: RS-01

RS-02			
Descripción:	El sistema de información permitirá a un usuario previamente registrado autenticarse en la aplicación.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-02, RUC-03 RUR-01, RUR-07		

TABLA 49: RS-02

RS-03			
Descripción:	EL sistema de información permitirá al usuario acceder al modo de cifrado de imágenes.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-04, RUC-17		

TABLA 50: RS-03

RS-04			
Descripción:	El sistema de información permitirá al usuario acceder al modo de descifrado de imágenes.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-05, RUC-17		

TABLA 51: RS-04

RS-05			
Descripción:	El sistema de información permitirá al usuario abrir una imagen para cifrar o descifrar, habiendo elegido este previamente el modo de uso de la aplicación.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-06, RUC-17, RUR-02		

TABLA 52: RS-05

RS-06			
Descripción:	El sistema de información permitirá al usuario añadir la condición de cifrado por contraseña.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-07, RUC-17, RUC-18, RUR-03, RUR-05		

TABLA 53: RS-06

RS-07			
Descripción:	El sistema de información permitirá al usuario añadir la condición de cifrado por el punto de acceso a WI-FI actual.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-08, RUC-17, RUC-18, RUR-07		

TABLA 54: RS-07

RS-08			
Descripción:	El sistema de información permitirá al usuario añadir la condición de cifrado por un punto acceso a WI-FI introducido manualmente.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-08, RUC-17, RUC-18, RUR-07		

TABLA 55: RS-08

RS-09			
Descripción:	El sistema de información permitirá al usuario añadir la condición de cifrado una fecha y una hora concretas.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-10, RUC-17, RUC-18		

TABLA 56: RS-09

RS-10			
Descripción:	El sistema de información permitirá al usuario añadir la condición de cifrado el nombre de usuario de Topoos.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-11, RUC-17. RUC-18, RUR-07		

TABLA 57: RS-10

RS-11			
Descripción:	El sistema de información permitirá al usuario cifrar una imagen con las condiciones de cifrado elegidas.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☐Alta	☑Baja	
Requisitos:	RUC-07, RUC-08, RUC-09, RUC-10, RUC-11, RUC-14, RUR-06		

TABLA 58: RS-11

RS-12			
Descripción:	El sistema de información permitirá al usuario almacenar una imagen cifrada en el terminal.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-13, RUC-17.		

TABLA 59: RS-12

RS-13			
Descripción:	El sistema de información permitirá al usuario enviar una imagen cifrada con una aplicación de terceros de su elección.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-12, RUC-17		

TABLA 60: RS-13

RS-14			
Descripción:	El sistema de información permitirá al usuario descifrar una imagen si se cumplen las condiciones de cifrado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☐Alta	☑Baja	
Requisitos:	RUC-15, RUR-04, RUR-07, RUR-08, RUR-09, RUR-10		

TABLA 61: RS-14

RS-15			
Descripción:	El sistema de información permitirá almacenar las imágenes descifradas		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-16		

TABLA 62: RS-15

RS-16			
Descripción:	El sistema de información permitirá al usuario comprobar los detalles de la versión de la aplicación.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☐Necesario	☑Opcional	
Estabilidad:	☐Alta	☑Baja	
Requisitos:	–		

TABLA 63: RS-16

3.10.2. REQUISITOS DE OPERACIÓN

RS-17			
Descripción:	La aplicación permitirá acceder al modo cifrar desde la pantalla principal		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-04, RUC-17		

TABLA 64: RS-17

RS-18			
Descripción:	Acceder al modo descifrar desde la pantalla principal		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-05, RUC-17		

TABLA 65: RS-18

RS-19			
Descripción:	El usuario podrá consultar los detalles de la versión desde la pantalla principal		
Prioridad:	☐Alta	☐Media	☑Baja
Necesidad:	☐Necesario	☑Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-17		

TABLA 66: RS-19

RS-20			
Descripción:	El sistema permitirá la navegación por las interfaces de los distintos módulos de la aplicación para poder corregir o modificar la información. Para ello se habilitará un botón "Atrás", así como los correspondientes botones para avanzar en el proceso de cifrado o descifrado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☐Alta	☑Baja	
Requisitos:	RUC-17, RUC-18		

TABLA 67: RS-20

RS-21			
Descripción:	El botón "Atrás" estará presente en todas las interfaces de la aplicación a excepción de la pantalla principal.		
Prioridad:	☐Alta	☑Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-18		

TABLA 68: RS-21

RS-22			
Descripción:	El sistema mostrará botones para elegir los distintos tipos de cifrado. Dichos botones serán: - Contraseña - WI-FI ID - Fecha / Hora - Usuario		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☐Alta	☑Baja	
Requisitos:	RUC-04, RUC-07, RUC-08, RUC-09, RUC-10, RUC-11, RUC-17		

TABLA 69: RS-22

RS-23			
Descripción:	Junto a todos los botones, se colocará un icono con un mensaje explicativo o de ayuda desplegable.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-17		

TABLA 70: RS-23

RS-24			
Descripción:	El sistema de información mostrará un botón para seleccionar la aplicación de terceros a la que se desea mandar la imagen cifrada.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-12, RUC-17		

TABLA 71: RS-24

RS-25			
Descripción:	El sistema de información deberá mandar la imagen a la aplicación de terceros que se haya seleccionado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-12		

TABLA 72: RS-25

RS-26			
Descripción:	El sistema de información mostrará un aviso en caso de que algún campo necesario no se haya completado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-17, RUR-05		

TABLA 73: RS-26

RS-27			
Descripción:	La aplicación permitirá al usuario borrar las imágenes originales inmediatamente después de su cifrado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-14		

TABLA 74: RS-27

3.10.3. REQUISITOS DE INTERFAZ

RS-28			
Descripción:	La aplicación podrá ser llamada de manera externa cuando se seleccione desde el menú contextual de Android "abrir con" sobre una imagen.		
Prioridad:	☑Alta	☐Media	☑Baja
Necesidad:	☐Necesario	☑Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-12		

TABLA 75: RS-28

RS-29			
Descripción:	La aplicación siempre se utilizará en posición vertical (portrait)		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-17		

TABLA 76: RS-29

RS-30			
Descripción:	Los colores utilizados para los fondos y los textos deben tener un contraste moderadamente alto.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-17		

TABLA 77: RS-30

3.10.4. REQUISITOS DE RENDIMIENTO

RS-31			
Descripción:	El sistema de información debe permitir al usuario poder seleccionar imágenes de cualquier resolución y tamaño.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-06, RUR-02		

TABLA 78: RS-31

RS-32			
Descripción:	El sistema realizará una acción de cifrado o de descifrado siempre en un tiempo menor de un minuto.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-06, RUR-08		

TABLA 79: RS-32

3.10.5. REQUISITOS DE RECURSOS

RS-33			
Descripción:	La aplicación debe disponer de conexión a Internet para realizar el log-in, descifrar una imagen con la fecha y la hora, nombre de usuario o WI-FI ID como clave y para comprobar si existen nuevas actualizaciones que están disponibles.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-07		

TABLA 80: RS-33

RS-34			
Descripción:	La aplicación funcionará en sistemas operativos Android a partir de la versión de firmware 2.3.0.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	–		

TABLA 81: RS-34

RS-35			
Descripción:	El terminal deberá contar con la memoria RAM suficiente para poder utilizar el sistema.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	–		

TABLA 82: RS-35

3.10.6. REQUISITOS DE SEGURIDAD

RS-36			
Descripción:	La aplicación utilizará algoritmos de cifrado seguros, es decir, no vulnerables computacionalmente.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-06		

TABLA 83: RS-36

RS-37			
Descripción:	La aplicación contará con una carpeta oculta, no accesible por la galería de Android, en caso de que el usuario pretenda almacenar sus imágenes descifradas.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-13		

TABLA 84: RS-38

RS-38			
Descripción:	La aplicación no mostrará las contraseñas cuando se estén introduciendo.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-03, RUR-04		

TABLA 85: RS-39

RS-39			
Descripción:	La aplicación garantizará que únicamente se realice el descifrado de una imagen si se cumplen todas las condiciones de cifrado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-09		

TABLA 86: RS-39

RS-40			
Descripción:	La aplicación deberá cumplir la Ley Orgánica de Protección de Datos.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	–		

TABLA 87: RS-40

3.10.7. REQUISITOS DE VERIFICACIÓN

RS-41			
Descripción:	La aplicación tiene que controlar los errores que se puedan producir durante su funcionamiento y proporcionar los mensajes adecuados en el caso de que ocurran.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☐Alta	☑Baja	
Requisitos:	RUC-03		

TABLA 88: RS-41

RS-42			
Descripción:	La aplicación mostrará mensajes informativos si no se cumple una condición de descifrado.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-10		

TABLA 89: RS-42

RS-43			
Descripción:	La aplicación comprobará que se haya abierto una imagen antes de permitir cifrarla o descifrarla.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUC-06		

TABLA 90: RS-43

RS-44			
Descripción:	La aplicación comprobará que la imagen tenga la información suficiente como para ser cifrada o descifrada.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-02		

TABLA 91: RS-44

RS-45			
Descripción:	La aplicación comprobará para cada uno de los campos de cifrado elegidos, que no estén vacíos.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-05, RUR-06		

TABLA 92: RS-45

RS-46			
Descripción:	La aplicación comprobará que la contraseña sea suficientemente segura.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-05		

TABLA 93: RS-46

RS-47			
Descripción:	Antes de descifrar, la aplicación comprobará que condiciones de cifrado se han utilizado para cifrar dicha imagen.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	-		

TABLA 94: RS-47

RS-48			
Descripción:	La aplicación comprobara que se cumplan las condiciones de descifrado antes de permitir descifrar una imagen.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-09		

TABLA 95: RS-48

RS-49			
Descripción:	Para descifrar, la contraseña introducida debe coincidir con la contraseña de cifrado, solo en caso de que se haya cifrado con contraseña.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-09		

TABLA 96: RS-49

RS-50			
Descripción:	Para descifrar, el punto de acceso a WI-FI utilizado debe coincidir con el introducido en el modo de cifrado, solo en caso de que se haya cifrado con WI-FI ID.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-07, RUR-09		

TABLA 97: RS-50

RS-51			
Descripción:	Para descifrar, la contraseña introducida debe coincidir con la contraseña de cifrado, solo en caso de que se haya cifrado con contraseña.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-09		

TABLA 98: RS-51

RS-52			
Descripción:	Para descifrar, la hora, obtenida de internet debe ser igual o mayor que la introducida en el momento de cifrado, solo en caso de que se haya cifrado con fecha y hora.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-07, RUR-09		

TABLA 99: RS-52

RS-53			
Descripción:	Para descifrar, el usuario introducido debe coincidir con el introducido en el modo de cifrado, solo en caso de que se haya cifrado con usuario.		
Prioridad:	☑Alta	☐Media	☐Baja
Necesidad:	☑Necesario	☐Opcional	
Estabilidad:	☑Alta	☐Baja	
Requisitos:	RUR-07, RUR-09		

TABLA 100: RS-53

3.11. ANÁLISIS DE CLASES

En este apartado se definirán las clases que serán necesarias para desarrollar Image Cipher. Dichas clases, nacen a partir de los requisitos y casos de uso definidos anteriormente.

3.11.1. IDENTIFICACIÓN DE LAS CLASES

Dado que la aplicación se diseñará para que funcione con el sistema operativo móvil Android, tenemos que tener en cuenta que necesitaremos una clase por cada conjunto de interfaces de la aplicación. Esto se justifica sabiendo que Android utiliza una actividad por pantalla mostrada. Desde el punto de vista del análisis podemos entender que una actividad se refleja perfectamente en una clase. Por ello, las clases que se van a definir son las siguientes:

- **Pantalla principal:** En esta actividad se elegirá qué uso se le dará a la aplicación (cifrar o descifrar) pasando así a la correspondiente actividad. Asimismo, se incluirá un acceso para que el usuario pueda ver las novedades de la versión.

- **Cifrar:** Esta actividad contendrá las interfaces y las funcionalidades que se usarán para cifrar una imagen.
- **Descifrar:** Esta actividad contendrá la interfaz en la que se mostrará la imagen descifrada, en caso de que el proceso haya sido el correcto, o los mensajes de error correspondientes en caso contrario.
- **Pantalla de confirmación**: Esta actividad contiene un mensaje que informa al usuario de que el proceso ha salido correctamente y le dará a elegir qué hacer con la imagen obtenida (guardar o enviar).
- **Util:** Esta clase no se trata de una actividad sino de código no visible para el usuario. En ella se contendrán los métodos para el cifrado y descifrado de imágenes. También incluyen funcionalidades especiales sobre las cuales se apoyan las demás clases de la aplicación.

3.11.2. ESPECIFICACIÓN DE LAS FUNCIONES DE CADA CLASE

Una vez identificadas en el apartado anterior, se pasará a definir los métodos más relevantes de cada una:

- **Pantalla Principal:**
  - irACifrar: Se encargará de ejecutar el código que permita el acceso a la clase Cifrar.
  - irADescifrar: Se encargará de ejecutar el código que permita el acceso a la clase Descifrar.
  - irAVersión: Se encargará de ejecutar el código que permita el acceso a las novedades de la versión.
- **Cifrar:**
  - AbrirImagen: Se encargará de ejecutar el código que permita abrir una imagen con el fin de cifrarla.
  - AñadirContraseña: Facilitará al usuario la interfaz para introducir una contraseña.
  - AñadirWifiID: Facilitará al usuario la interfaz para introducir una WIFI-ID.
  - AñadirFechaHora: Facilitará al usuario la interfaz para introducir una fecha y una hora concreta.
  - AñadirUsuario: Facilitará al usuario la interfaz para introducir un usuario determinado.
  - ConfirmarCifrar: Se encargará de ejecutar el código que permita la confirmación de las condiciones y acceda cifre la imagen mediante la clase Util.
- **Descifrar:**
  - Descifrar: este método leerá las condiciones asociadas a la imagen cifrada e informará al usuario de las condiciones no cumplidas. En caso de que éstas no existan, invocará a la clase Util para descifrar la imagen.
  - MostrarImagen: Mostrará la imagen obtenida por la clase Util, producto del proceso de descifrado.
- **Pantalla de Confirmación:**
  - MostrarConfirmación: Este método mostrará un mensaje indicando al usuario que todo el proceso (cifrado o descifrado) ha salido correctamente.

- GuardarImagen: Se encargará de almacenar la imagen obtenida en la memoria del teléfono.
- EnviarImagen: Se encargará de ejecutar el código necesario para que el usuario pueda seleccionar la aplicación de terceros mediante la cual quiera enviar la imagen.

- **Util:**
  - Cifrar: Este método se encarga de reunir todas las condiciones que el usuario ha introducido, así como la imagen, y después, la cifra.
  - Descifrar: Este método se encarga de reunir todas las condiciones que el usuario ha introducido, así como la imagen, y después, la descifra.
  - ConsultarFecha: Consulta, a la hora de descifrar, la fecha real de internet, y la pasa a la hora española para poder compararla con la condición de fecha y hora de cifrado.

Hay que recordar que estos métodos no son los métodos Java que se implementarán en las fases posteriores, sino las funcionalidades más importantes de cada clase. Los métodos que se implementarán se describirán más detalladamente en la fase de diseño de la aplicación.

3.11.3. -DIAGRAMA DE CLASES

A continuación, puesto que ya se han definido las clases que se van a utilizar, el diagrama de clases correspondiente:

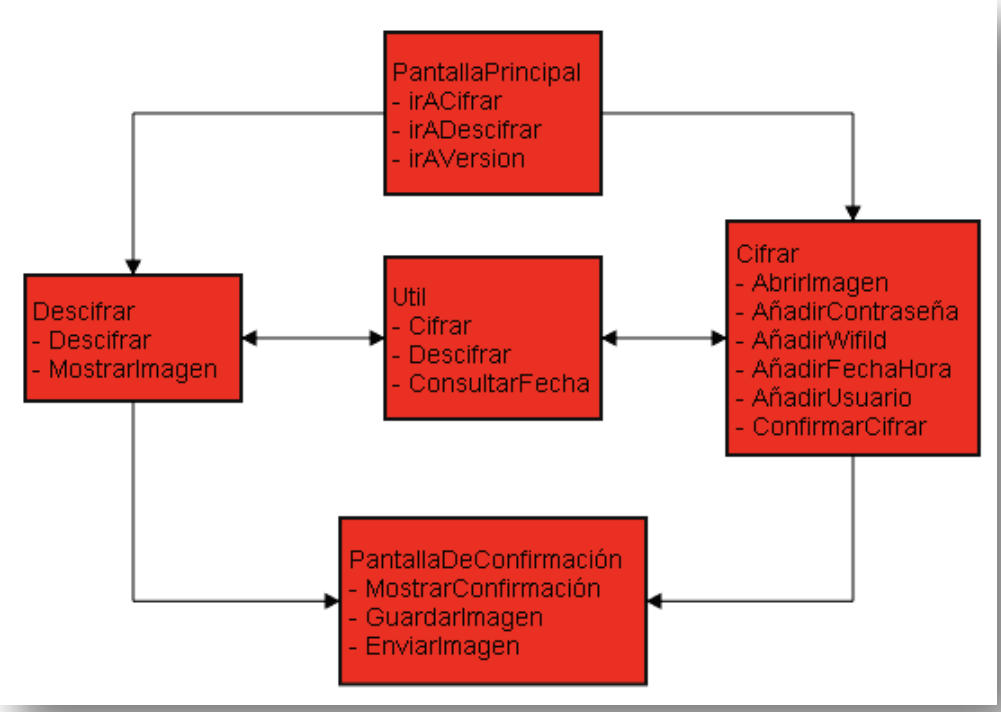


ILUSTRACIÓN 22: DIAGRAMA DE CLASES

Se trata de un diagrama muy sencillo, además es muy similar al flujo de ejecución de la propia aplicación. Vemos que desde PantallaPrincipal, podemos acceder a Cifrar o Descifrar. Cada una de éstas últimas, se apoyará en la clase Util, que es la que realmente llevará peso en cuanto a implementación. Util ayudará a Cifrar y Descifrar en la realización de sus tareas. Una vez finalizadas dichas tareas, se pasará la clase PantallaDeConfirmación. Esta clase, en función de la tarea realizada, mostrará unas opciones u otras, pero siempre tendrá disponible el mensaje de confirmación al usuario de que la operación ha sido satisfactoria. En caso de que no haya sido así, son las clases anteriores (Cifrar y Descifrar) las que mostrarán el mensaje de error correspondiente

4. DISEÑO

En esta fase se dará forma a la solución del problema planteado en el Análisis. Se deberán justificar todas las decisiones tomadas para tal fin.

Del mismo modo que en la fase anterior, se llevará un proceso incremental, es decir, se comenzará describiendo y definiendo la aplicación en rasgos generales para más tarde profundizar en ella dividiéndola en subsistemas y módulos.

4.2. ARQUITECTURA DEL SISTEMA

Para definir de manera efectiva la arquitectura, se ha decidido dividir el sistema de información en capas que se van comunicando con las capas inferiores. El siguiente gráfico muestra la arquitectura escogida:

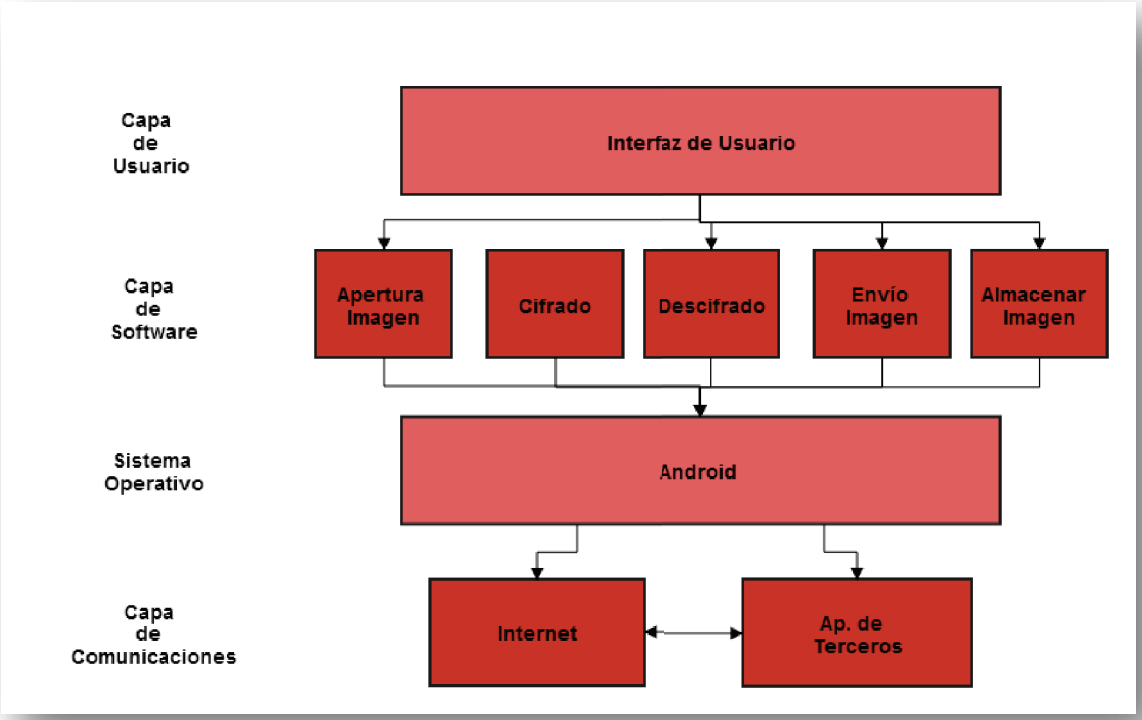


ILUSTRACIÓN 23: ARQUITECTURA DEL SISTEMA

Como se ve en el gráfico anterior, la arquitectura contará con cuatro capas horizontales. La capa que interactúa con el usuario, la capa de software, el sistema operativo Android y la capa de comunicaciones. Cada una de las capas únicamente se comunica con la que tiene inmediatamente debajo.

La capa de interfaces es con la que el usuario interactúa. Se comunica con los módulos de la capa de software que son los que contienen las funcionalidades de la aplicación. La capa de software, a su vez, se comunicará con el sistema operativo para ejecutar dichas funcionalidades. Android actuará como motor para la ejecución de las funcionalidades, pero también hace de intermediario con la capa de comunicaciones, que facilitarán la realización de las funcionalidades se la capa de software que tengan dependencias de internet o de aplicaciones externas.

Hay que tener en cuenta que las aplicaciones de terceros también pueden contar con esta dependencia de internet para poder funcionar, tales como aplicaciones de correo electrónico o mensajería.

A partir de la capa de Interfaz de Usuario, podemos acceder a los distintos módulos de la capa de Software, inmediatamente más baja. Los datos que necesita esta capa para realizar sus funciones son facilitados por el usuario en la capa anterior. Debido a esta estrecha relación, hay que prestar especial cuidado al diseño de la capa de Software.

En primer lugar, encontramos el módulo de Apertura de Imagen. Este módulo se encargará de abrir las imágenes seleccionadas por el usuario hacerlas manejables para la aplicación.

Después nos centramos en el módulo de Cifrado. Éste, si la imagen es válida, juntará las condiciones introducidas por el usuario y junto con los datos de la imagen, la cifrará. Después dará a elegir al usuario que quiere hacer con la imagen (almacenarla o enviarla)

Seguidamente, el módulo de Descifrado, tratará de reunir las condiciones necesarias con el fin de descifrar la imagen. Únicamente descifrará la imagen si se cumplen todas las condiciones introducidas. Posteriormente mostrará la imagen y dará al usuario la opción de almacenar la imagen.

También encontraremos en la parte final de la ejecución de la aplicación, que el usuario deberá elegir entre uno de los módulos de Envío o Almacenar Imagen. Cada uno de éstos se encargará de realizar los cambios y modificaciones pertinentes para realizar un envío a una aplicación de terceros o garantizar su almacenamiento respectivamente. En una iteración de la aplicación sólo se ejecutará uno de los dos módulos.

Todos los módulos de la capa de Software se comunican con la capa de Sistema Operativo, inmediatamente inferior. Esta capa tan sólo consiste en el sistema operativo Android del terminal del usuario. Esta capa, se encarga de ejecutar las llamadas al sistema que se realicen.

La última capa, de la Comunicaciones, posee dos módulos. El primero de ellos es Internet. Se encargará de reunir los datos de capas anteriores que requieran conexión en línea, como por ejemplo, la obtención de la fecha real para el descifrado de la imagen. Es evidente que para que este módulo funcione correctamente, se necesitará un terminal que cuente con acceso a internet en el momento en que se ejecute.

El segundo, Aplicaciones de Terceros, consiste en el conjunto de aplicaciones instaladas en el terminal del usuario que sean compatibles con Image Cipher. Es posible que una o varias de estas aplicaciones también dependan, como se ha mencionado anteriormente, del módulo Internet, es decir, es probable que existan vías de comunicación entre ambos módulos.

Dada la peligrosidad de esta relación es necesario utilizar las funciones nativas de Android para la comunicación entre estas aplicaciones e Image Cipher.

Vemos que los módulos más importantes serán los de cifrado y descifrado en la capa de Software. Serán por ello, los primeros en diseñarse y también en los que se llevará a cabo su implementación.

Como se puede deducir, los módulos principales del sistema son los que tengan que ver con la generación de códigos y con la lectura de ellos. Estos módulos serían los primeros en diseñarse e implementarse, pero al depender directamente de la apertura de imágenes, se dará prioridad a este último.

4.3. SUBSISTEMAS

Una vez hemos definido la arquitectura que se va a diseñar, propuesta en el apartado anterior y fruto del proceso de análisis llevado a cabo, se va a optar por dividir el sistema de información en subsistemas para favorecer la manejabilidad a la hora de realizar el diseño.

Es muy conveniente que para el sistema que se está diseñando, los subsistemas coincidan con los módulos planteados en el apartado de la arquitectura ya que frecuentemente, los subsistemas se reconocen por las funcionalidades que ofrecen. Dichas funcionalidades, en conjunto, suelen tener una intención común, de ahí que se haya tomado esta decisión.

Los subsistemas que se van a plantear y diseñar, son pues, Apertura de Imágenes, Cifrado y Descifrado

4.3.1. SUBSISTEMA DE APERTURA DE IMÁGENES

El objetivo del módulo de apertura de imágenes es extraer una imagen válida de la galería de Android y realizar con ella los cambios pertinentes para que los demás módulos puedan manejarla. La imagen original, podrá estar en varios formatos, algo que deberemos tener en cuenta si queremos realizar esta tarea de forma satisfactoria.

A modo de ejemplo se mostrará la interacción con los módulos y capas adyacentes para la realización de las funciones de la apertura de imágenes es la siguiente:

- La orden de selección de imágenes de la galería de Android viene dada por la capa de Interfaz. En ella, se invoca a la función nativa del sistema operativo para desplegar dicha aplicación.
- Una vez en la galería, el propio usuario se desplaza a través de todos los ficheros de imagen que dispone en el terminal.
- Cuando el usuario selecciona una imagen, el sistema operativo devuelve el control al módulo de apertura, con la imagen que ha seleccionado el usuario
- El módulo, en primer lugar, comprueba que la imagen seleccionada sea válida, realizando las comprobaciones pertinentes, definidas en los requisitos del sistema.

- Posteriormente, el subsistema da formato a la imagen obtenida, haciéndola manejable para los demás módulos de la aplicación. En este proceso, también se hará un reconocimiento del formato de origen de la imagen, que se almacenará para el cifrado de la imagen.

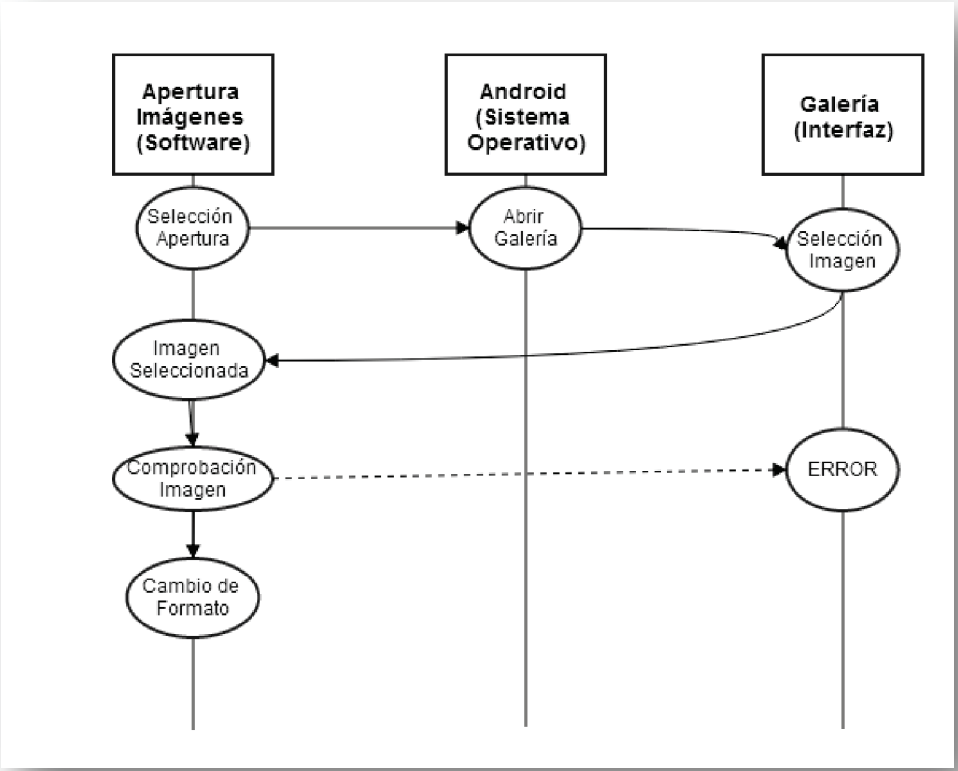


ILUSTRACIÓN 24: SUBSISTEMA DE APERTURA DE IMÁGENES

En el gráfico anterior, se ve el flujo de ejecución que se llevará a cabo para este subsistema de información.

4.3.2. SUBSISTEMA DE CIFRADO

Una vez tenemos la imagen en un formato que la aplicación puede manejar, podemos elegir cifrarla. El subsistema de cifrado consta de varias etapas internas que son necesarias para el cifrado de la imagen en sí. Los componentes de este subsistema son los siguientes:

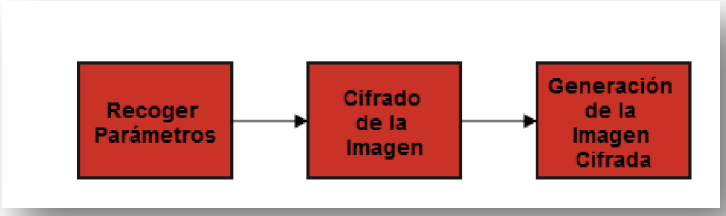


ILUSTRACIÓN 25: SUBSISTEMA DE CIFRADO

Dada la complejidad de cada componente, se ha optado por explicar su contenido en los siguientes apartados.

4.3.2.1. RECOGER PARÁMETROS

La aplicación dará al usuario a elegir entre varias opciones que se deberán cumplir para descifrar la imagen que se va a cifrar a continuación. Las condiciones de cifrado o condiciones de descifrado, son una serie de parámetros que el propio usuario introduce.

Los parámetros posibles son:

- Contraseña: Será la contraseña de cifrado. En caso de no introducirse, la aplicación cifrará la imagen con una contraseña por defecto
- WI-FI ID: Se trata del identificador del punto de acceso a WI-FI.
- Fecha / Hora: A partir de la cual podremos descifrar la imagen.
- Usuario: El identificador de usuario de Topoos.

Todos los parámetros son opcionales. Aunque el usuario no introduzca ninguno de ellos, el cifrado de la imagen se seguirá llevando a cabo, aunque se manera más débil.

Para cada uno de los parámetros se realizarán sendas comprobaciones, concretamente las siguientes:

- Para el parámetro contraseña:
  - Si la contraseña está en blanco, se descarta la contraseña.
  - Que la contraseña no contenga caracteres no permitidos.
  - Que la contraseña sea suficientemente larga.
  - Que la contraseña cuenta con una minúscula.
  - Que la contraseña cuenta con una mayúscula.
  - Que la contraseña cuenta con un símbolo.
- Para el parámetro WI-FI ID:
  - Si se introduce vacío, se descarta el parámetro.
  - Que el WIFI-ID no contenga caracteres no permitidos.
  - Que el WIFI-ID tenga el formato correcto.
  - Si se escoge usar el WIFI-ID actual (auto-detectado) se descarta el introducido por el usuario.
- Para el parámetro Fecha / Hora:
  - Que la fecha y la hora no sean anteriores a la actual.
  - En caso de que la fecha y la hora introducidas sean muy cercanas a la actual, se lanzará un aviso.

- Para el parámetro Usuario:
    - Que exista dicho usuario en la base de datos de la aplicación en la plataforma Topoos.

Si una de las comprobaciones no se cumple, se lanzará un mensaje de error al usuario, a través de la capa de Interfaz.

Los parámetros introducidos irán siendo almacenados temporalmente en la aplicación, hasta el momento en el que se sobre-escriban o se complete el proceso de cifrado de la imagen.

4.3.2.2. CIFRADO DE LA IMAGEN

La imagen obtenida, se cifrará utilizando el algoritmo de cifrado AES-256. La contraseña de cifrado derivará directamente de la contraseña introducida por el usuario, o en su lugar, por la contraseña introducida por defecto por la aplicación. La clave de cifrado se obtiene con el algoritmo PBKDF2 (Password-Based Key Derivation Function 2). Se trata de un potente algoritmo que realizará funciones hash. La derivación de clave dispara el tiempo de cómputo necesario para un ataque de fuerza bruta contra la propia clave.

Para llevar a cabo este proceso de derivación se concatena en primer lugar un *salt* aleatorio a la contraseña introducida por el usuario, y se realizará el proceso PBKDF2 un número concreto de iteraciones. Posteriormente, se añadirán caracteres de relleno con el algoritmo PKCS5 para fortalecerla aún más.

Posteriormente, la clave obtenida se introducirá junto con la imagen, previamente formateada en el cifrador AES en modo CBC (Cipher-Block Chaining). Con esto obtendremos un vector de inicialización con un tamaño de bloque adecuado.

Finalmente, se concatena la imagen fuertemente cifrada, con el salt y el vector de inicialización obtenido para poder descifrar la imagen posteriormente.

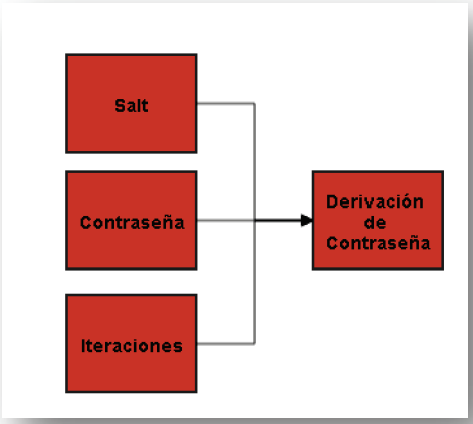


ILUSTRACIÓN 26: DERIVACIÓN DE CONTRASEÑA DE CIFRADO

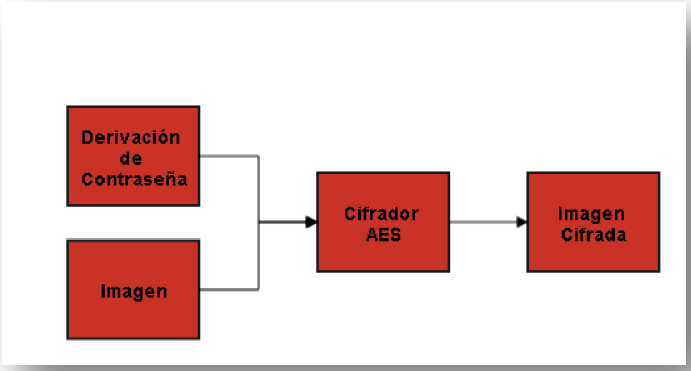


ILUSTRACIÓN 27: PARÁMETROS DE ENTRADA DEL CIFRADOR

4.3.2.3. GENERACIÓN DE LA IMAGEN CIFRADA

Por tanto, la interacción entre los distintos componentes es la siguiente:

- En primer lugar, se mostrarán las distintas interfaces al usuario para que éste escoja los parámetros para el posterior cifrado de la imagen. Dichos parámetros, no son otros que las condiciones de cifrado que se han mencionado a lo largo del proceso de análisis. La imagen, se ha obtenido como producto del subsistema de Apertura de Imágenes.
- El usuario elegirá uno, varios o ningún parámetro para el cifrado de la imagen. Estos parámetros se irán almacenando para construir la lista de condiciones.
- Una vez escogidas y confirmadas las condiciones de cifrado, se pasará a inicializar el componente de cifrado.

4.3.2.3.1. CREACIÓN DE TOKENS

Una vez cifrada la imagen, se necesitará adjuntar las claves para ver si una imagen es descifrable o no. Dado que las condiciones de descifrado pueden llegar a ser largas, se ha elegido utilizar unos tokens de identificación para cada una de ellas.

El fin de esto es que a la hora de descifrar, la aplicación sea capaz de leerlos rápidamente antes de comenzar el descifrado de toda la imagen. Dicho proceso puede ser moderadamente costoso computacionalmente hablando, por lo que es más conveniente realizar la comprobación antes de iniciarlo.

En primer lugar se utilizará una marca para comprobar si la imagen ha sido cifrada o no con la clave por defecto.

Después existirá otro token para comprobar si se ha añadido la condición WI-FI ID. En caso afirmativo, se adjuntará el WI-FI ID.

Posteriormente se hará lo propio con la fecha / hora y con el usuario de Topoos.

El resultado será una cadena de caracteres, lista para adjuntarse a su vez con la imagen cifrada.

4.3.2.3.2. CONCATENAR LOS DATOS

Para finalizar el proceso de cifrado, se juntarán las condiciones de descifrado con la propia imagen cifrada. El orden de este proceso se ha escogido de la siguiente manera: en primer lugar, se colocarán las condiciones de cifrado, seguidas del grueso del fichero, que es la propia imagen cifrada. Esto se debe a que para ahorrar tiempo de cómputo a la hora de descifrar, no se descifrará directamente la imagen, sino que previamente habrá que realizar las comprobaciones correspondientes, para ver si las condiciones se cumplen.

Asimismo, las condiciones, no deben ir en texto plano, sino que la cadena que las almacena, pasará por el propio módulo de cifrado con una contraseña determinada por la aplicación. Una vez tengamos las condiciones cifradas, se adjuntará el grueso del fichero con la imagen cifrada. De esta forma, cuando se llegue al proceso de descifrar, la aplicación descifrará en primer lugar las condiciones y comprobará que se cumplan todas. En caso contrario, no descifrará el resto del fichero de imagen.

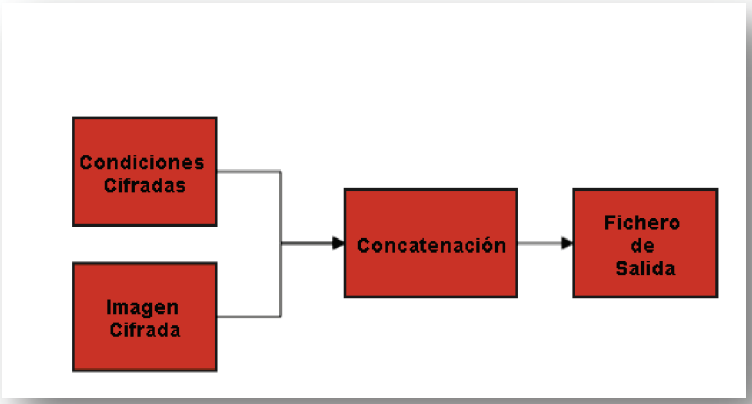


ILUSTRACIÓN 28: CONCATENACIÓN DE DATOS

4.3.3. SUBSISTEMA DE DESCIFRADO

Una vez obtenemos una imagen cifrada correctamente y pretendemos descifrarla, tenemos que tener en cuenta la estructura del fichero que se ha generado al cifrar. Sabemos que existe un bloque de datos con las condiciones de descifrado de la imagen y que, según lo que se ha decidido anteriormente, debemos descifrar primero para luego comprobar dichas condiciones. Así, en caso de que alguna condición no se cumpliera ahorraríamos tiempo y recursos, computacionalmente hablando.

Por ello, los componentes que formarán este subsistema serán los siguientes:

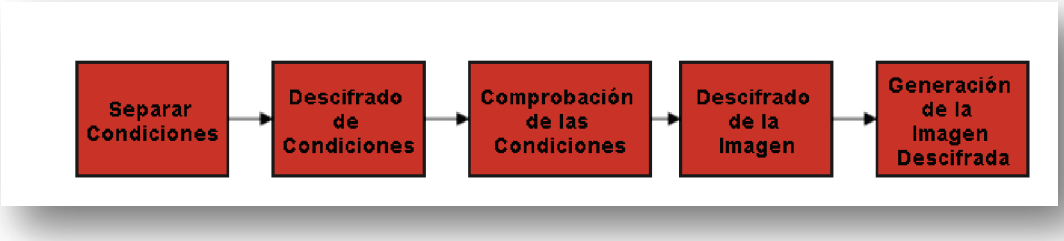


ILUSTRACIÓN 29: SUBSISTEMA DE DESCIFRADO

Dada la complejidad de cada componente, se ha optado por explicar su contenido en los siguientes apartados:

4.3.3.1. SEPARACIÓN DE CONDICIONES

En primer lugar, tal y como se ha mencionado en el apartado anterior, será necesario dividir el fichero para aislar las condiciones con el fin de trabajar con ellas.

Dado que la cadena original de las condiciones de cifrado será de tamaño variable, sabemos que la cadena cifrada de datos, es poco tolerante a cambios de tamaño. Por ello se ha optado por una división dinámica, es decir, por asumir que las condiciones de descifrado serán de tamaño variable.

Teniendo esto en cuenta, en el subsistema de cifrado, en el proceso de *join* de los datos, se incluirá una marca de separación única, de manera que cuando este componente entre en ejecución, sabrá de forma rápida hasta qué punto del fichero abarca la cadena cifrada de condiciones y en qué punto comienza el grueso de la imagen.

Se realizará un proceso de lectura secuencial del fichero, puesto que no sabemos cuál será este punto. No obstante sabemos con certeza que se encontrará relativamente cerca del comienzo del conjunto de datos que forma el fichero de imagen cifrada. Una vez encontrada la marca de separación, este componente se encargará de dividir la imagen y las condiciones, para poder trabajar sobre ellas por separado.

4.3.3.2. DESCIFRADO Y COMPROBACIÓN DE CONDICIONES

Una vez obtenemos las condiciones de descifrado, podremos trabajar con ellas de forma previa a hacerlo con el resto de la imagen.

4.3.3.2.1. DESCIFRADO DE CONDICIONES

En primer lugar, debemos descifrar los datos obtenidos. Una de las mayores utilidades de los cifradores de bloque, como se contaba en la introducción de este documento, es la facilidad que tienen estos algoritmos de cifrado para ser reutilizables a la hora de descifrar datos. Es extremadamente sencillo descifrar las condiciones (como se hará también con la imagen posteriormente), únicamente debemos seguir los pasos del cifrador en sentido contrario.

Se utilizará el mismo algoritmo, con la misma derivación de contraseña y el mismo salt. Evidentemente conocemos el algoritmo pero no ocurre lo mismo con la derivación o el salt. Estos dos últimos datos, deberán venir siempre como caracteres especiales en la cadena cifrada de datos. Conociendo esos tres elementos necesarios, solo tenemos que centrarnos en la contraseña.

Dado que para este caso, y por motivos de rendimiento y usabilidad se han cifrado las condiciones con una contraseña conocida (únicamente por la aplicación), asumiremos que conocemos también la contraseña.

Con todos los datos necesarios, ya obtenemos la entrada para el algoritmo de descifrado:

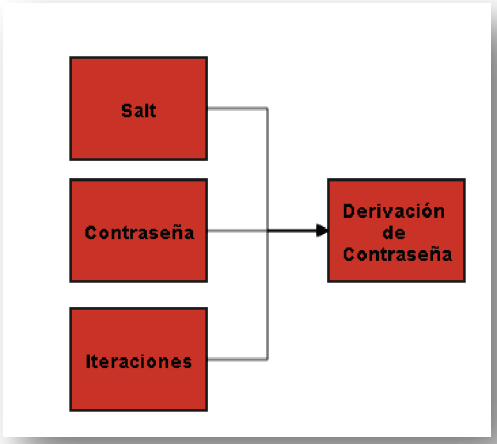


ILUSTRACIÓN 30: DERIVACIÓN DE CONTRASEÑA DE DESCIFRADO

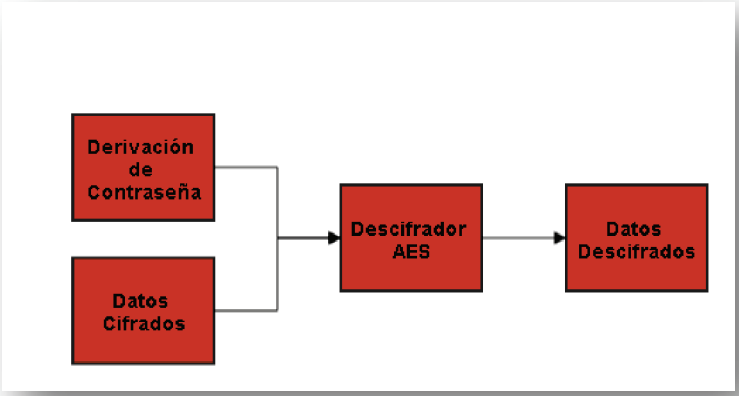


ILUSTRACIÓN 31: PARÁMETROS DE ENTRADA DEL DESCIFRADOR

4.3.3.2.2. COMPROBACIÓN DE CONDICIONES

Sólo cuando hayamos obtenido la cadena de caracteres descifrada con las condiciones de descifrado, podremos pasar a comprobar si se cumplen o no.

Recordamos que las condiciones pueden no existir, como por ejemplo, en el caso de que el usuario no haya introducido una fecha concreta a partir de la cual la imagen sea descifrable, no obtendremos una fecha que comprobar.

De nuevo por razones de rendimiento, se analizarán las condiciones en orden de coste computacional, esto es, en primer lugar las comprobaciones más sencillas, seguidas de las que conlleven un proceso más largo. Las condiciones más costosas serán las que requieran acceso a internet, interpretación de datos o cambio de tipos. Para todas ellas, antes de

realizar las comprobaciones pertinentes, se realizará, evidentemente, la comprobación de si existe esa condición en concreto o no.

La primera condición que se analizará será la de contraseña de cifrado. Esta condición, si no está marcada como vacía, significa que el usuario que cifró introdujo una contraseña en este proceso. Si es así, la aplicación deberá realizar una llamada a la capa de usuario para que abra una interfaz para que el usuario introduzca la contraseña para descifrar la imagen. Es evidente que si el usuario no introduce una contraseña que se corresponda con la que tenemos como condición, se mostrará un mensaje de error a tal efecto y no se descifrará la imagen, ni siquiera se comprobarán más condiciones.

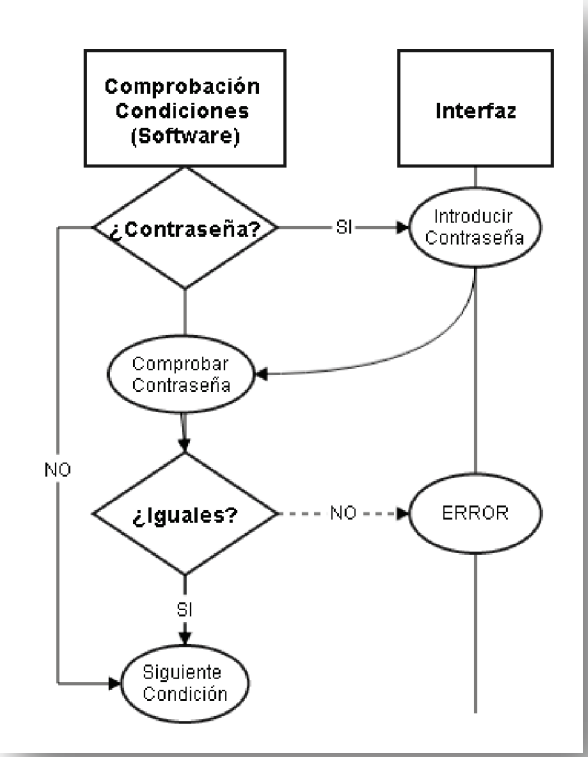


ILUSTRACIÓN 32: COMPROBACIÓN DE CONTRASEÑA

En segundo lugar, se analizará la condición de WIFI-ID. En caso de que no esté vacía, se hará una comprobación previa, por si el usuario no estuviese conectado a un punto de acceso WI-FI. Es importante mantener al usuario informado, por ello, se realizará una llamada a la capa de usuario para que muestre un mensaje informativo en caso de que sea necesario habilitar la conexión WI-FI del terminal, dando al usuario a elegir a que punto de acceso quiere conectarse. Una vez se detecte que el dispositivo está conectado, se realizará la comprobación entre el SSID del punto de acceso actual y el que está marcado como condición.

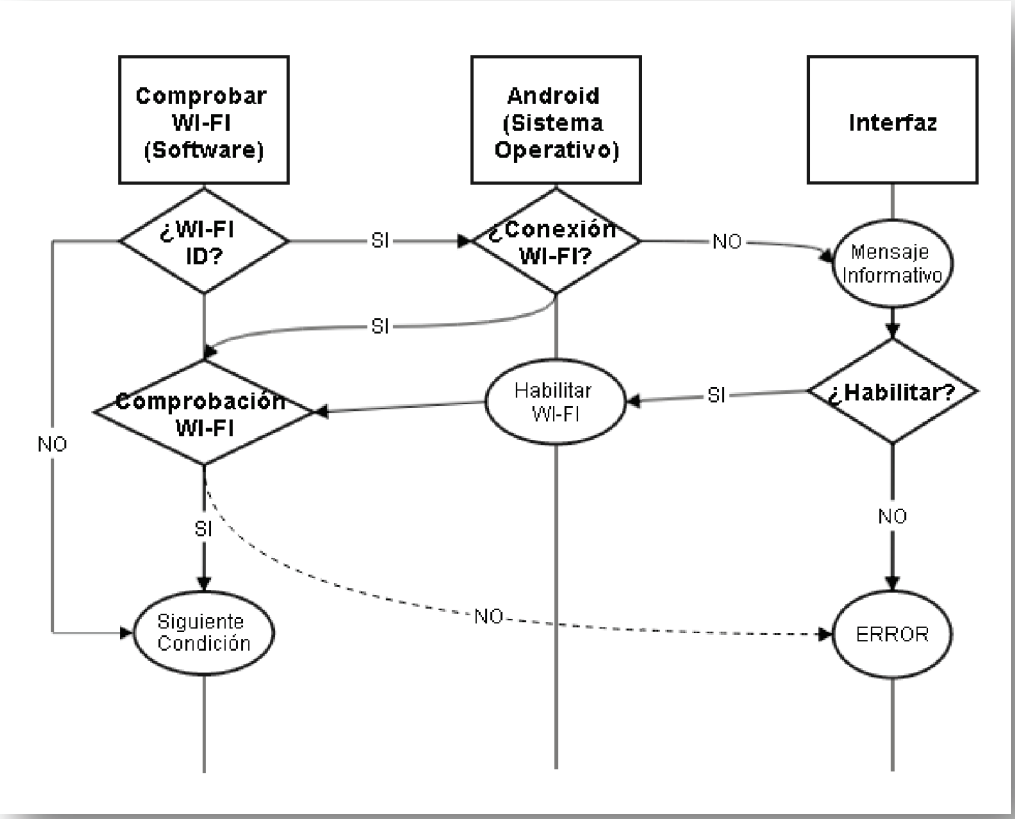


ILUSTRACIÓN 33: COMPROBACIÓN DE WI-FI ID

Para las siguientes comprobaciones será necesario el acceso a internet. Si el mensaje que se determinaba en el apartado anterior no se ha mostrado. Se informará al usuario de que se va a acceder a internet para realizar las comprobaciones, siempre dándole la opción de terminar el proceso antes de que esta operación se lleve a cabo.

En el caso de la condición de fecha y hora, recordamos que se necesita conexión a internet por motivos de seguridad. Si se realizara la comprobación usando la fecha y la hora del sistema operativo del dispositivo, un usuario podría alterar esta fecha y hora de manera muy sencilla desde los ajustes del terminal, saltándose la comprobación. Es por eso por lo que necesitamos obtener la hora exacta de un servidor remoto, de manera que no se pueda modificar. La aplicación por ello, realizará una petición a este servidor. La respuesta obtenida puede no ser la adecuada, es decir, puede que no se encuentre en un formato adecuado para la comparación con la fecha / hora almacenada como condición. Además, hay que cerciorarse, de que la fecha y la hora, se correspondan de manera lógica con la hora real actual. Por ejemplo, si el servidor se encontrara en un país extranjero con distinto huso horario, será necesario modificar la hora para que se corresponda con la hora donde se encuentre el usuario. Para este proyecto, se va a asumir que el usuario se encontrará siempre en suelo español y en zona peninsular. Una vez nos aseguramos de que la hora sea correcta, pasamos a darle un formato de manera que podamos compararla con la condición.

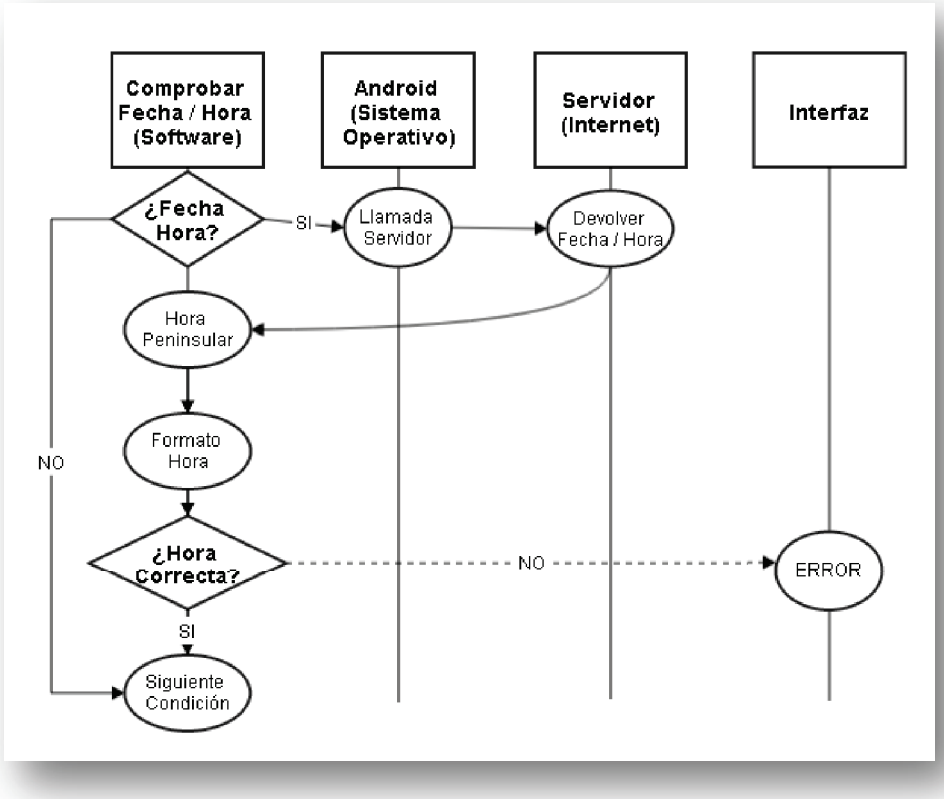


ILUSTRACIÓN 34: COMPROBACIÓN DE FECHA / HORA

Algo similar ocurre con la última condición a comprobar. El usuario de Topoos introducido también debe coincidir para continuar el proceso de descifrado de la imagen. Para obtener el nombre del usuario de la aplicación, también se realiza una llamada a un servidor, en esta ocasión de Topoos, que devuelve dicho usuario. La aplicación no necesita el identificador del usuario actual, sino el nombre, que será el que el usuario que cifró la imagen habría introducido como parámetro a la hora de cifrar. En el caso de que la llamada se realice de manera satisfactoria, la aplicación realizará una comparación de los nombres obtenido y actual, prestando atención en ignorar las mayúsculas y minúsculas, para no dar pie a "falsos negativos".

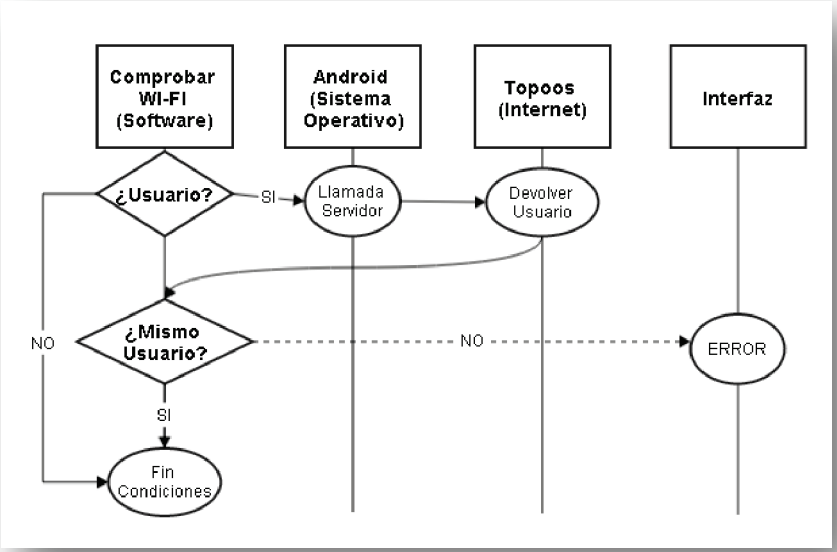


ILUSTRACIÓN 35: COMPROBACIÓN DE USUARIO

Una vez realizada la comprobación de todas las condiciones de descifrado, se puede comenzar el proceso de descifrado de la imagen en sí. Hay que tener en cuenta, no obstante, que los parámetros comprobados en este componente, se pueden desechar para liberar memoria, dado que ciertos terminales son limitados en este aspecto. Como excepción a este proceso de borrado tenemos que marcar la contraseña si existe, o comunicar al componente de descifrado que debe utilizar la contraseña por defecto de la aplicación para llevar a cabo su función.

4.3.3.3. DESCIFRADO DE LA IMAGEN

Con las condiciones validadas por un lado y la imagen cifrada por otro, llegamos al punto en el que nos es posible comenzar a descifrar la parte de imagen que contenía el fichero de entrada para este subsistema.

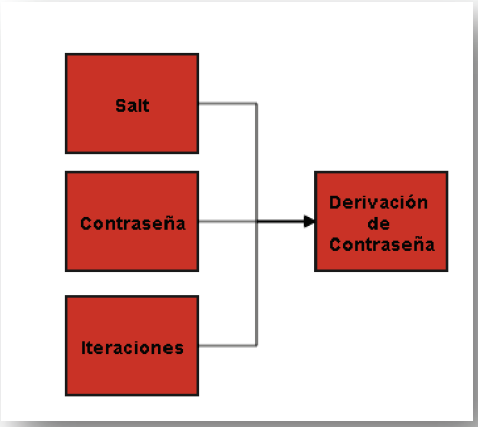
En este componente, debemos tener en cuenta la contraseña final con la que se va a descifrar la imagen. Recordamos que en caso de que el usuario que cifró en primer la imagen original no introdujo una contraseña, la aplicación proporcionaba una por defecto, con una seguridad aceptable y de una longitud moderada. Si el usuario introdujo la contraseña, en este punto ya habrá sido validada, es decir, habrá pasado de ser una condición como las citadas en el apartado anterior a ser realmente la contraseña necesaria para el algoritmo de descifrado. En cualquiera de estos casos, tenemos la contraseña de descifrado ya disponible para realizar las operaciones necesarias.

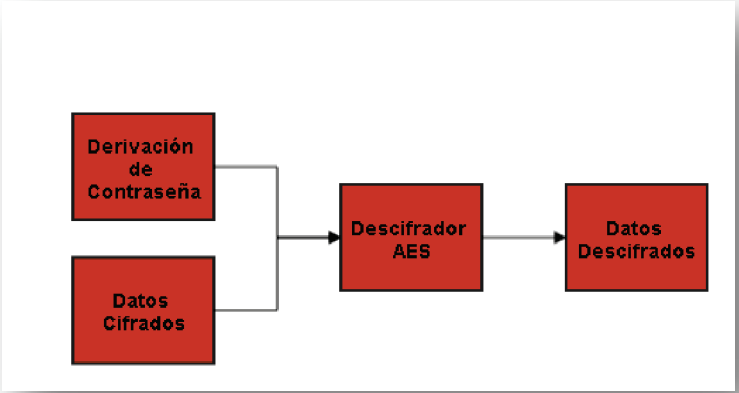
Del mismo modo que se comentaba en el apartado de descifrado de condiciones, se necesitan más parámetros de entrada para el cifrador. Dichos parámetros son los mismos,

se requerirá el mismo *salt* y la misma derivación de la clave, de otra manera el cifrador bien dará un error, o bien producirá un resultado no satisfactorio.

Estos dos parámetros, se obtienen de la misma forma que en el proceso de descifrado de las condiciones, es decir, se obtienen de caracteres especiales de la cadena cifrada de datos. La aplicación deberá saber cómo localizarlos para que sirvan de entrada al cifrador.

Por esto, vemos una vez más que el esquema para el proceso de descifrado de la imagen es el siguiente:





El resultado, los datos descifrados, necesitan una última modificación para que la imagen descifrada sea legible. Dicha modificación se determina en el apartado siguiente.

4.3.3.4. GENERACIÓN DE IMAGEN DESCIFRADA

Cuando se ejecuta esta componente, la imagen ya se ha descifrado de manera satisfactoria. No obstante, se encontrará almacenada en memoria en el dispositivo y se deberá hacer legible.

Para convertir los datos en imagen de nuevo, siguiendo el proceso inverso del subsistema de apertura de imágenes, tenemos que prestar especial atención a las cabeceras de fichero. Como se comentaba en apartados previos, las imágenes que puede manejar un smartphone tienen varios formatos. Cada uno de ellos cuenta con una extensión correspondiente a su contenido y codificación. Si formamos una imagen con un formato y una extensión que no son compatibles el resultado nunca será satisfactorio.

Por ello, las imágenes que ya hemos descifrado, deben mantener una extensión afín a su contenido, es decir, una imagen que siendo descifrada tenga una compresión JPEG, deberá tener la extensión correspondiente ".jpeg", una con PNG, deberá tener la extensión ".png" y así sucesivamente. La aplicación, por tanto, deberá detectar que tipo de compresión tiene la imagen resultado para asignarle una extensión correcta antes de enviar la imagen o almacenarla, según decida el usuario.

4.3.4. SUBSISTEMA DE ENVÍO DE IMÁGENES

Como último paso, el usuario podrá elegir enviar su imagen cifrada con una aplicación externa. Evidentemente, el dispositivo deberá tener disponible y operativa una aplicación compatible con el envío de imágenes. Las aplicaciones compatibles, aparecerán en el menú de Image Cipher denominado "Enviar" que precisamente lista estas aplicaciones.

Algunas de dichas aplicaciones pueden ser:

- Email: Tanto la aplicación Email como Gmail, que permiten al usuario del dispositivo gestionar y vincular varias cuentas de correo electrónico a fin de recibir mensajes nuevos, así como para crearlos. Ambas aplicaciones son capaces de adjuntar ficheros, entre ellos archivos de imagen para que sean enviados junto con el correo electrónico. Cuando se selecciona esta opción en "Enviar a", se abrirá un nuevo correo y la imagen tratada se adjuntará como código a él. Por tanto no hace falta adjuntarla desde la galería de Android, aunque también es posible.
- Dropbox: Dropbox es una plataforma para almacenamiento online. Permite compartir ficheros con otros usuarios de esta aplicación. Es lo que se conoce vulgarmente como almacenamiento en la nube. Dropbox permitirá, también en su versión para Smartphone, subir todo tipo de archivos a una cuenta, moverlos entre las carpetas, descargarlos o eliminarlos. Si el usuario selecciona Dropbox en el menú "Enviar a", podrá subir la imagen a su cuenta de Dropbox, eligiendo la carpeta que prefiera.
- Google Drive: Drive es una aplicación gestionada por Google muy similar a Dropbox. Permite el almacenamiento online de ficheros, pero esta vez, se suben a cuentas de Google. Del mismo modo, el usuario que seleccione esta opción en el menú "Enviar a", podrá subir la imagen a su cuenta, eligiendo en que carpeta lo hará.

- Whatsapp: Es una aplicación de mensajería instantánea. Además de enviar mensajes (es la aplicación con más flujo de mensajes del mundo en la actualidad), también se pueden enviar ficheros multimedia tales como imágenes, videos o audio. Si el usuario elige esta opción en el menú "Enviar a", elegirá a que contacto o grupo de contactos enviará la imagen tratada con Image Cipher. Whatsapp solo permite el envío a contactos previamente registrados, por lo que el usuario deberá introducir el número de teléfono y guardar al contacto al que quiera enviarla de manera previa.
- Telegram: Se trata de una aplicación muy similar a Whatsapp. Recientemente es de las más descargadas ya que es gratuita, diferencia muy potente en comparación con su competidora, que cobra anualidades. La integración de esta aplicación funciona exactamente del mismo modo que en el caso de Whatsapp.
- Facebook: La aplicación móvil de la gran red social. La aplicación permite cambiar el estado del perfil autorizado en el terminal, así como todas las funcionalidades que permite la red social en su versión de "sobremesa", hacer comentarios o "like" en las publicaciones de nuestros contactos, subir fotos, y compartir videos u otros recursos de internet. Cuando el usuario seleccione "Enviar a" Facebook, podrá compartir la imagen en su espacio (denominado muro) de la cuenta autorizada del dispositivo. Facebook también permite restringir que usuarios o contactos verán la foto.
- Twitter: Se trata de otra plataforma que ofrece servicios de la red social. Permite publicar tweets, ver los tweets de otros usuarios, "seguir" a usuarios, etc. En definitiva, permite hacer lo mismo que permite la versión web de Twitter. Si el usuario elige Twitter en el menú "Enviar a", la imagen se adjuntará como código y se enviará y publicará en la plataforma.

Aunque estas son las formas más comunes de enviar una foto desde un terminal Android, existen muchas otras, tales como mensajes multimedia MMS, Bluetooth, etc. También hay que tener en cuenta de que irán surgiendo nuevas aplicaciones capaces de realizar esta función. Si una aplicación es compatible, podrá accederse a ellas desde el menú "Enviar a".

4.3.5. SUBSISTEMA DE ALMACENAMIENTO DE IMÁGENES

El usuario podrá elegir almacenar una imagen cuando haya completado el proceso de cifrado o descifrado con la aplicación que se va a desarrollar. El proceso de almacenamiento de imágenes es muy restrictivo por motivos de seguridad.

Hay que tener en cuenta que cuando se cifre una imagen, si se descifra, ya sea en el terminal propio o en el de otro usuario, almacenándose la imagen después, la imagen quedará accesible por otras aplicaciones, por lo cual la imagen podrá quedar comprometida.

El almacenamiento de imágenes cifradas podrá hacerse en una carpeta en la memoria del terminal cualquiera. No ocurrirá así con las imágenes descifradas por el motivo que se describe en el párrafo anterior. Si la galería de Android tiene acceso a la imagen, de manera muy sencilla, otra aplicación podría "compartirla", por ello se ha recomendado, siguiendo los casos de uso RUC-16 y RS-15, que el almacenamiento de imágenes descifradas se realice en una carpeta creada por la aplicación que se va a desarrollar que se ocultará a la aplicación de galería del sistema operativo del dispositivo.

Así, el subsistema de almacenamiento de imágenes descifradas, no permitirá al usuario elegir la ruta en la que almacenará sus ficheros recién obtenidos.

4.4. INTERFACES DE USUARIO

Una de las partes más importantes de una aplicación es su diseño de interfaces. En este caso, el diseño sencillo e intuitivo figura como uno de los requisitos de usuario más citados. A continuación se diseñan las interfaces que va a utilizar la aplicación.

4.4.1. ESTUDIO DE DISEÑO

Antes de realizar el diseño en sí, es necesario realizar un pequeño estudio para comprender los posibles problemas que se puedan causar a un usuario provocados por la interfaz. Asimismo, se debe comprender todo aspecto en el que se pueda facilitar el uso de la aplicación al usuario de Image Cipher mediante las interfaces diseñadas.

4.4.1.1. SENCILLEZ E INTUICIÓN

El requisito de usuario RUC-17 especifica que la navegación debe ser "sencilla e intuitiva" para poder cumplir este requisito, definido de forma ligeramente ambigua, debemos especificar lo que se pretende cumplir y definir de manera precisa ambos conceptos.

- Sencilla: Que no ofrezca dificultad al usuario final. Las interfaces deben de ser lo más sencillas posible, para que no confundir al usuario y para minimizar la curva de aprendizaje en el proceso de uso de la aplicación. [33]
- Intuitiva: Que permite la comprensión inmediata, sin apenas hace uso de razonamiento. Esto se traduce en que los movimientos sobre la pantalla que deberá realizar el usuario cuando utilice la aplicación deberán coincidir con lo que hará la media de los usuarios. [34]

Por tanto, para hacer la aplicación con interfaces sencillas, se minimizará el número de opciones posibles, así como los botones en las pantallas.

Para cumplir el requisito y que la aplicación sea intuitiva, es conveniente seguir una serie de recomendaciones. Las que se seguirán para el desarrollo de la presente aplicación serán las establecidas por Google. [35]. Entre ellas podemos encontrar las siguientes:

- El usuario debe saber en todo momento qué está haciendo con la aplicación. Si por ejemplo un usuario selecciona una opción concreta, esta deberá resaltarse momentáneamente para hacer saber al usuario que la aplicación se ha percatado de la propia acción del usuario. Esto se define como concepto de retroalimentación.
- Es recomendable el uso de colores primarios. Para la presente aplicación se utilizarán los colores y tonos facilitados por el entorno de programación y diseño de Android.

- Utilizar una fuente color y tamaño de texto concreta. Esta debe ser concisa, sencilla, amigable. Los textos deberán resaltar lo más importante poniéndolo al principio de cada texto y utilizar solamente el texto necesario.

Para la presente aplicación se utilizará el estándar de diseño "Holo". Holo cumple las recomendaciones mencionadas, además de poseer un contraste suficiente como para que la aplicación sea fácilmente legible ante luz solar indirecta (podrá utilizarse sin problemas al aire libre). En modo retrato (Portrait) la barra de estado superior será visible por el usuario en todo momento.

La tipografía utilizada por Holo se denomina "Roboto". Esta tipografía, utilizando el alto contraste del patrón de diseño, cumplirá las recomendaciones de Google.

Es preciso dar importancia al contraste en el diseño de las interfaces. Muchas de las aplicaciones disponibles en el mercado se usan con mucha dificultad cuando se está al aire libre, en especial, podemos referirnos a juegos para Smartphone que utilizan degradado de colores. Hacer al usuario final forzar la vista a la hora de utilizar la aplicación se ha considerado muy inapropiado.

4.4.1.2. VOLVER ATRÁS

Viendo el concepto de retroalimentación, y sabiendo que el proceso a seguir para cifrar una imagen mediante la aplicación es largo, se pretende no confundir al usuario obligándole a utilizar exclusivamente el botón "Back" de Android, sino también mostrar un botón "Atrás" que cumpla la misma función. De este modo el usuario será consciente de que puede volver a modificar parámetros del proceso.

El requisito de usuario RUC-18 define que el usuario debe ser capaz de volver atrás en todo momento. Esto quiere decir, será capaz de realizar un retorno a la pantalla anterior en caso de que pretenda modificar o eliminar algún parámetro tales como alguna condición o la propia imagen. El usuario deberá poder incluso cambiar de modo (cifrado o descifrado) en la misma ejecución de la aplicación.

Esto viene estrechamente entrelazado con el concepto de retroalimentación definido en el apartado anterior. El usuario deberá saber en qué estado del proceso de cifrado o descifrado se encuentra y qué pantallas quedan para completar el proceso.

El botón atrás deberá colocarse siempre en la misma posición de la pantalla. Para que la navegación se intuitiva, se colocará de manera que el usuario sepa inmediatamente donde se encontrará el botón de atrás y donde el botón "siguiente" que corresponda a pasar al siguiente estado de la ejecución.

Así pues, se determina que el botón atrás siempre se encontrará en la parte inferior izquierda de la pantalla, estando los botones de continuación en la parte inferior derecha.

4.4.2. PANTALLAS DE AUTENTICACIÓN

En primer lugar, la primera vez que el usuario accede a la aplicación se encontrará con una pantalla que únicamente le permita autenticarse o registrarse en la plataforma Topoos. Dicha pantalla será como la siguiente:

ILUSTRACIÓN 36: PANTALLA DE BIENVENIDA

Posteriormente, el usuario será redirigido a la página web de Topoos, donde deberá darse de alta para comenzar a usar la aplicación. El registro se realiza a través de un formulario parecido al siguiente, al que se accede a través de un texto en la ventana de autenticación:

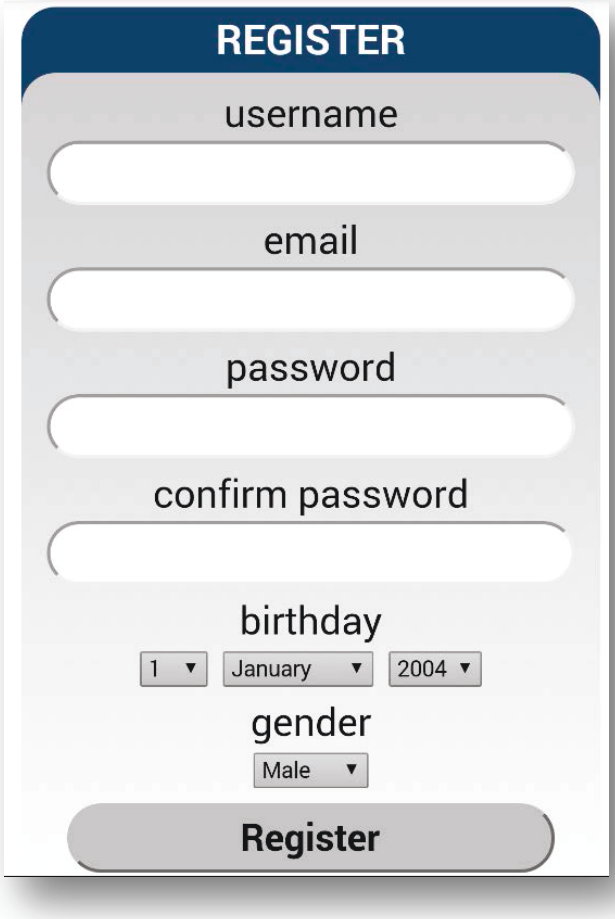


ILUSTRACIÓN 37: FORMULARIO DE REGISTRO

En el caso de que el usuario ya esté registrado, pero no autenticado, la pantalla que se mostrará será la siguiente:

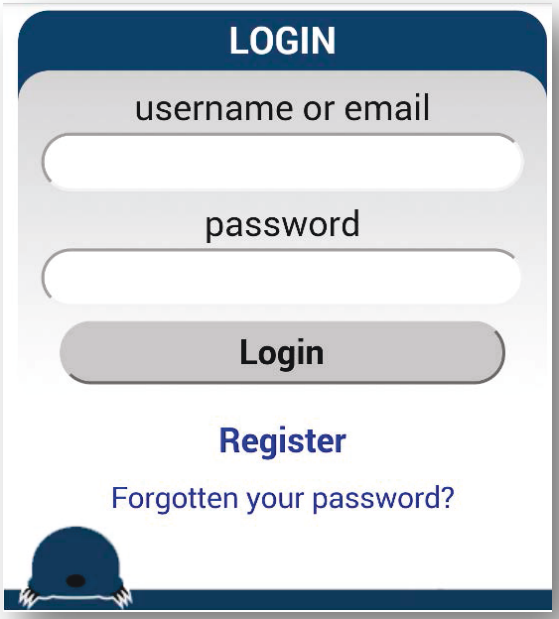


ILUSTRACIÓN 38: FORMULARIO DE AUTENTICACIÓN

4.4.3. PANTALLA PRINCIPAL

Salvo que la autenticación no se haya realizado, la aplicación mostrará su pantalla principal. En ella se podrá elegir entre los dos principales modos de ejecución:

ILUSTRACIÓN 39: PANTALLA PRINCIPAL

Dependiendo de la elección efectuada por el usuario se accederá a un módulo u a otro de la aplicación.

4.4.4. CIFRAR

En el momento en el que el usuario seleccione “Cifrar”, apareceremos en la siguiente pantalla:

ILUSTRACIÓN 40: SELECCIÓN DE IMAGEN (CIFRAR)

En esta interfaz vemos un claro ejemplo de cómo va a ser la interacción con la aplicación en cuanto a retroalimentación y flujo de ejecución.

- Se muestra un título que indica en que pantalla nos encontramos.
- Los botones están claramente diferenciados del texto plano.
- Los botones tendrán una función que los resalte cuando el usuario pulse sobre ellos.
- Los botones Atrás y Continuar están correctamente colocados.

La imagen estará visible para el usuario en la interfaz cuando la haya seleccionado.

Una vez el usuario haya pulsado en seleccionar la imagen se abrirá la aplicación de Galería de Android:

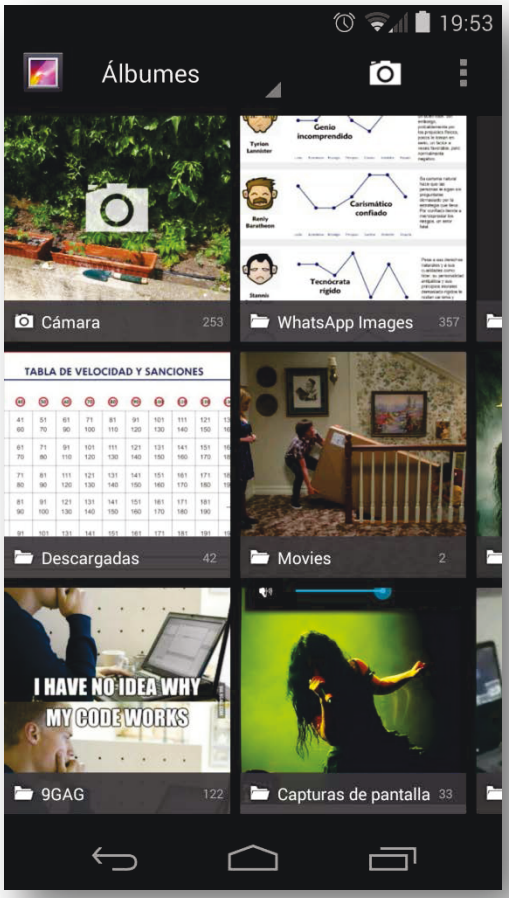


ILUSTRACIÓN 41: GALERÍA DE ANDROID

Cuando, habiendo seleccionado una imagen válida se pulse Continuar, el usuario verá la siguiente pantalla:

ILUSTRACIÓN 42: PANTALLA DE SELECCIÓN DE CONDICIONES

En esta pantalla encontramos el selector de las condiciones de cifrado para la imagen. Mediante el botón que corresponda el usuario pasará a las siguientes pantallas:

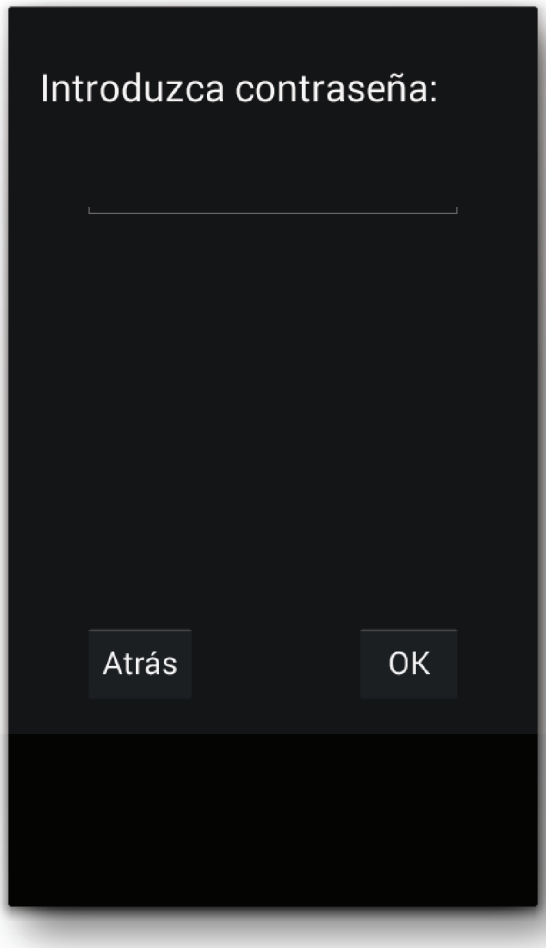


ILUSTRACIÓN 43: PANTALLA DE CONDICIÓN DE CONTRASEÑA

ILUSTRACIÓN 44: PANTALLA DE CONDICIÓN DE WI-FI ID

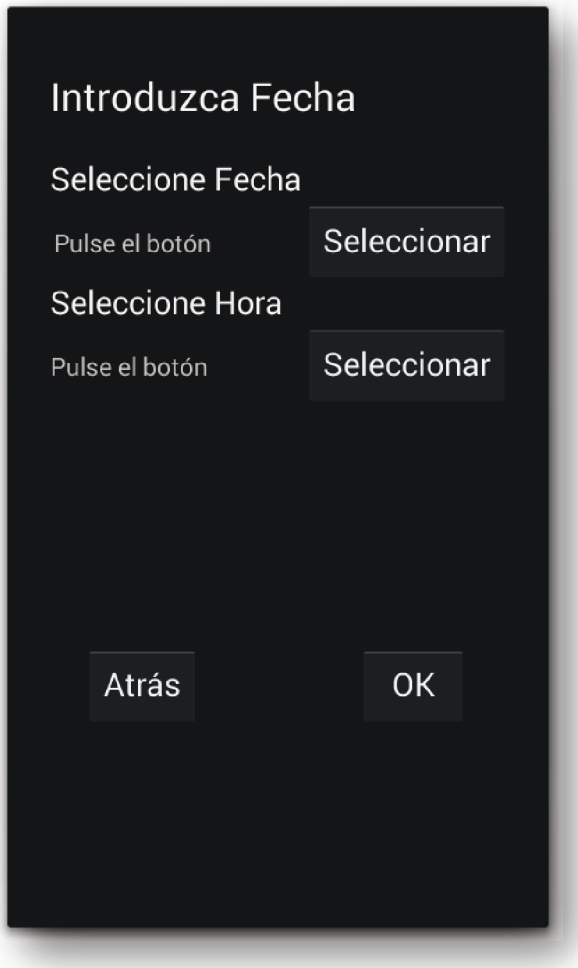


ILUSTRACIÓN 45: PANTALLA DE CONDICIÓN DE FEHCA / HORA

ILUSTRACIÓN 46: PANTALLA DE COINDICIÓN DE USUARIO

Cuando el usuario haya terminado de seleccionar las condiciones necesarias, se le pedirá confirmación para iniciar el proceso con la pantalla:

ILUSTRACIÓN 47: PANTALLA DE ELIMINACIÓN DE ORIGINAL

Vemos que es aquí donde aparece la opción del borrado de imágenes originales, para que la aplicación elimine la imagen que se está cifrando una vez acabado el proceso de cifrado.

Tras ella viene la pantalla en la que podemos ver cómo va el proceso:

ILUSTRACIÓN 48: PANTALLA DE PROGRESO (CIFRAR)

Cuando el proceso termina, se pasa a la pantalla de confirmación.

4.4.5. DESCIFRAR

Si el usuario selecciona descifrar en la pantalla principal, inmediatamente verá la siguiente pantalla:

ILUSTRACIÓN 49: PANTALLA DE SELECCIÓN DE IMAGEN (DESCIFRAR)

Esta pantalla es casi idéntica a la pantalla de selección de imagen del modo de cifrado (no se mostrará la imagen cifrada en la interfaz). Una vez haya seleccionado la imagen que el usuario quiera descifrar y pulse en continuar, el proceso será automático, salvo que la aplicación requiera contraseña. En este caso aparecerá la siguiente pantalla:

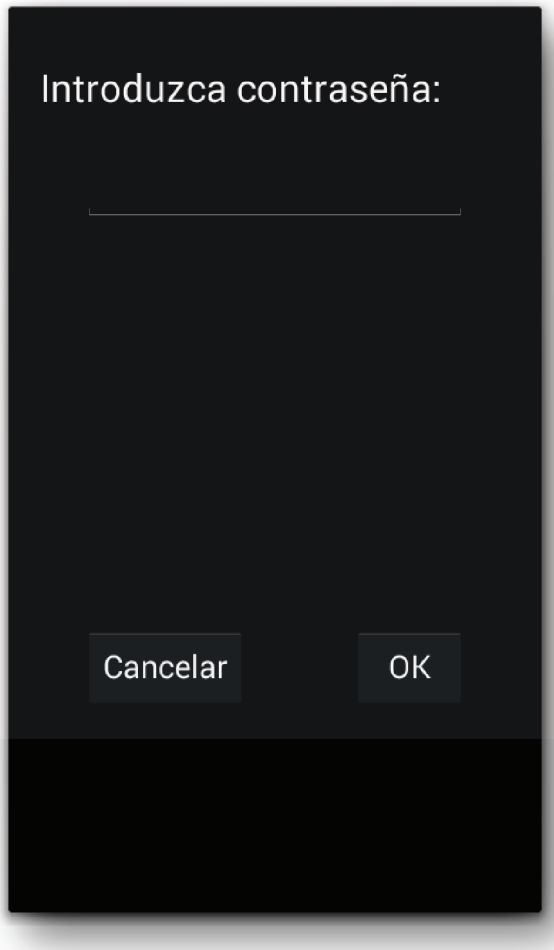


ILUSTRACIÓN 50: PANTALLA DE SOLICITUD DE CONTRASEÑA

Tanto si la contraseña es correcta como si no se requería contraseña, se iniciará el proceso de descifrado, y el usuario podrá ver cómo evoluciona:

ILUSTRACIÓN 51: PANTALLA DE PROGRESO (DESCIFRAR)

Una vez finalizado el proceso, se pasa a la pantalla de confirmación.

4.4.6. PANTALLA DE CONFIRMACIÓN

Como se mencionaba en la definición de subsistemas, la pantalla de confirmación se trata en realidad de un módulo que contiene varias pantallas o interfaces. Dependiendo de qué proceso acabe de terminar, aparecerán unas pantallas u otras.

Si el proceso que termina es el de cifrado aparecerá la siguiente interfaz:

ILUSTRACIÓN 52: PANTALLA DE CONFIRMACIÓN (CIFRAR)

En esta pantalla el usuario podrá elegir entre almacenar o enviar la imagen cifrada.

Para el caso en el que el usuario haya descifrado satisfactoriamente una imagen:

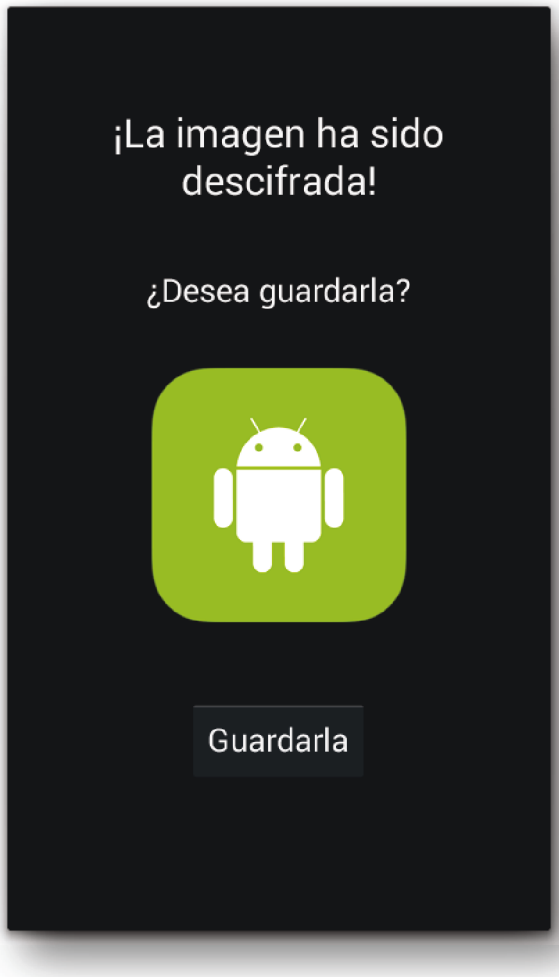


ILUSTRACIÓN 53: PANTALLA DE CONFIRMACIÓN (DESCIFRAR)

El usuario podrá elegir guardar la imagen o descartarla mediante el uso del botón “Back”.

4.4.7. OTRAS INTERFACES

Si en la pantalla principal el usuario interacciona con la aplicación con el fin de consultar las novedades de la versión de la aplicación, se desplegará una interfaz con dichas novedades:

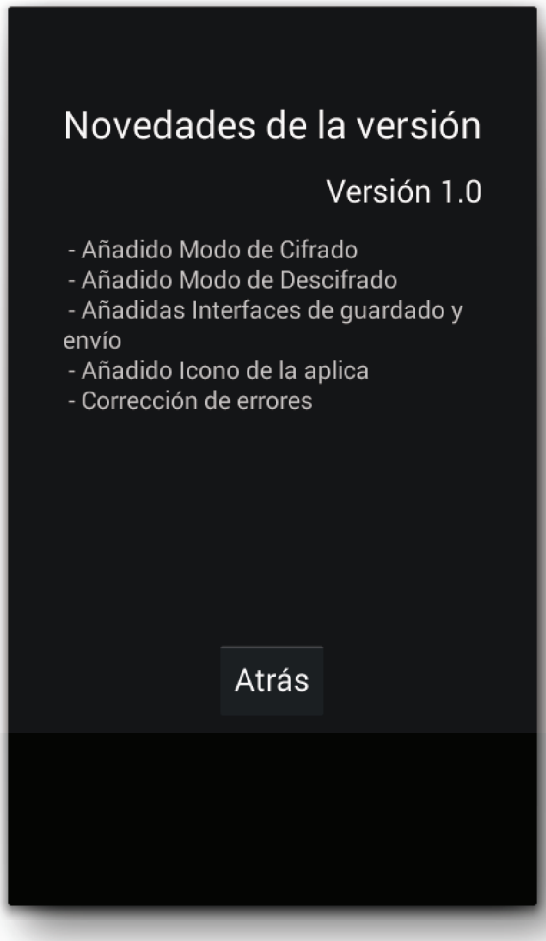


ILUSTRACIÓN 54: PANTALLA DE NOVEDADES

Asimismo, se utilizará una interfaz determinada para los errores controlados que puedan surgir en la aplicación. Dicha interfaz será similar a la siguiente:

ILUSTRACIÓN 55: PANTALLA DE ERROR

En esta pantalla, se ha pretendido mostrar al usuario que algo no ha ido bien. Para hacer hincapié en ello, se ha cambiado radicalmente el color de la interfaz. Se mostrará un mensaje con el correspondiente error controlado y se permitirá al usuario salir de la misma mediante el botón “Back”

5. IMPLEMENTACIÓN

Una vez finalizados los procesos de Análisis y Diseño del sistema de información, podemos comenzar el proceso de implementación del sistema. Dicho proceso, consistirá en realizar la programación de la propia aplicación Image Cipher. La implementación también se deberá dividir en distintos módulos dada la complejidad de los mismos.

Antes, no obstante, se deben definir varios conceptos sobre la programación en Android. Aunque este lenguaje es prácticamente idéntico a Java, existen sutiles matices a tener en cuenta. A continuación se enumeran los conceptos:

- Activity: Se trata de un tipo de clase Java normalmente asociadas a una pantalla. En esa pantalla, el usuario podrá interactuar con los elementos visibles en la interfaz y declarados en la clase. La totalidad de muchas aplicaciones consiste en varias actividades entrelazadas entre sí. Las actividades poseen un ciclo de vida, desde su creación, hasta su eliminación, pasando por pausa, retorno etc. Los ciclos de visa se definen con métodos "on" tales como onCreate, onResume…
- Intent: Se trata de una intención de hacer algo con la aplicación, es decir, una acción definida con anterioridad que puede ejecutarse de forma posterior. Los intents más comunes son los que en primer lugar definen una activity y después se ejecutan para pasar de una a otra. Sin embargo también pueden utilizarse para invocar servicios del sistema operativo, como llamadas, sms o abrir el navegador.
- Dialog: Se trata de una ventana que aparece en una interfaz a modo de pop-up. Puede ser con mensajes informativos o de error, o puede tratarse de formularios simples, existen de varios tipos, normalmente incorporan selectores como checkboxes o radio-buttons
- Thread: Hilo de ejecución. Son elementos que se crean para realizar una tarea en segundo plano, es decir, de forma paralela y normalmente transparente al usuario. Es útil para la llamada a servicios web. Hay que tener en cuenta que no es posible introducir datos en el hilo principal de la aplicación, por los consecuentes errores que provocaría.
- Handler: Se trata de la solución al problema comentado en el punto anterior. Puesto que no es posible modificar datos del hilo de ejecución principal, en caso de que queramos obtener los datos que hay en un hilo secundario de ejecución, no podremos hacerlo sin este componente. Él se encargará de encapsular un objeto con los datos que necesitamos en el hilo principal para que así se pueda acceder a los datos que necesitemos.

ILUSTRACIÓN 56: ESQUEMA DE HANDLER

5.2. MÓDULO DE PANTALLA PRINCIPAL

Tal y como se comentaba en el apartado de Diseño, cuando el usuario arranca la aplicación en su terminal, aparecería en la interfaz que se ha definido anteriormente como "Pantalla Principal", en la que podrá elegir el modo de ejecución de la aplicación. Sin embargo, siguiendo las pautas establecidas en los requisitos del Análisis, únicamente los usuarios autenticados en la plataforma Topoos podrán utilizar la aplicación en cualquiera de sus modos.

5.2.1. AUTENTICACIÓN Y REGISTRO

Por ello es necesario facilitar un servicio al usuario para este mismo fin. Es en este punto cuando realmente se integrará el framework de Topoos en la aplicación, siendo éste el responsable de la gestión de usuarios autenticados o registrados.

El proceso de registro o autenticado (*Sign Up* o *Log In* respectivamente) se realizará mediante la misma clase Java, SignUpActivity.java. Dicha clase, comenzará comprobando si el usuario posee un token de autenticación válido para la plataforma Topoos en su terminal. Si es así, la aplicación automáticamente dirigirá al usuario a la propia pantalla principal.

La autenticación por token, consiste en que el usuario almacene, casi siempre de manera inconsciente (es un proceso que normalmente se realiza de manera transparente al usuario) datos de log in en un fichero del dispositivo. El fichero consiste en una cadena de caracteres especiales con esos datos, además de parámetros de verificación, y fechas de caducidad, que se añaden por evidentes motivos de seguridad.

Si el token no es válido o no existe, el usuario deberá realizar cualquiera de los dos procesos de forma no automática, para ello, en primer lugar se cargará la interfaz correspondiente especificada en el apartado de Interfaces de Usuario del presente documento.

Según el caso, el usuario deberá rellenar uno u otro formulario para la acción que desee realizar (sign up o log in). Cuando el proceso haya acabado satisfactoriamente, se almacenará el token en el dispositivo y se redirigirá al usuario a la pantalla principal de la aplicación.

5.2.2. PANTALLA PRINCIPAL

Una vez el usuario se haya autenticado mediante los procesos indicados en el apartado anterior, se mostrará la pantalla principal y su correspondiente interfaz. En ella, el usuario podrá elegir el modo con el que quiere utilizar la aplicación o bien consultar las novedades de la versión instalada. La nueva clase WelcomeActivity.java se encargará de inicializar dicha interfaz, así como sus componentes (imágenes y botones).

Dependiendo de la elección del usuario, se ejecutará un módulo u otro.

5.3. MÓDULO DE APERTURA DE IMÁGENES

Si bien el usuario escoge un determinado modo de ejecución en la pantalla principal, no obstante, en primer lugar, deberá escoger la imagen que quiere cifrar. Para ello se hará uso de una nueva clase Java, ImageSelectorActivity.java. Esta clase formará parte del producto obtenido en la fase de Diseño, en el módulo homónimo.

En la ImageSelectorActivity, en primer lugar se mostrará la interfaz, correspondiente a la pantalla de selección de imágenes que se comentaba en los apartados de cifrado y descifrado del apartado de Interfaces de Usuario del presente documento.

Una vez inicializada la interfaz, se comprueba si el dispositivo tiene almacenamiento, es decir, si dispone de una tarjeta SD o si por el contrario tiene las imágenes almacenadas en la memoria interna. Esta comprobación es necesaria puesto que existen modelos de smartphone que no necesitan (y no dan soporte) a almacenamiento de este tipo.

Después, en la interfaz, el usuario deberá pulsar sobre el botón para seleccionar la imagen de la galería. Tal y como se describe en el diseño, el sistema operativo abrirá dicha aplicación, permitiendo al usuario seleccionar la imagen que pretende cifrar o descifrar.

La imagen seleccionada es almacenada en la aplicación. Para este fin, debemos crear una clase denominada como DTO (Data Transfer Object). Dicha clase, además de almacenar la imagen en sí, contiene más parámetros, tales como el formato utilizado, dimensiones, etc. Es importante que en una clase de tipo DTO, no existan más métodos que los necesarios para su creación y el acceso a sus atributos, de manera que cuando creemos un objeto de un DTO únicamente podamos trabajar con ellos mediante el almacenamiento y la modificación de sus parámetros.

Si bien la aplicación de Galería devuelve la imagen seleccionada en formato de mapa de bits (clase Bitmap), para la modificación de la imagen (cifrado o descifrado) es más conveniente el uso de un vector de Bytes. Aunque el hecho de manejar objetos de tipo ByteArray es costoso computacionalmente para el terminal, resulta muy efectivo y a la vez cómodo para realizar operaciones criptográficas, de ahí que se haya determinado el uso de esta clase para tratar las imágenes mientras se cifran o descifran.

Además, una vez obtenida la imagen seleccionada en formato ByteArray, podemos ver las cabeceras de los ficheros, necesarias para poder determinar la compresión o el formato a la hora de almacenar la imagen final. Asimismo podemos verificar si las dimensiones o el tamaño de la imagen seleccionada es adecuada para la aplicación.

Una vez realizados estos procesos, se permite al usuario continuar la ejecución de cifrado o descifrado mediante el botón “Continuar”. Se incluirá el DTO de la imagen cuando se realice la llamada a la siguiente actividad.

5.4. MÓDULO CIFRAR

Si el usuario había escogido usar Image Cipher en modo de cifrado, se ejecutará este módulo. Este apartado se corresponde con el producto del diseño de Subsistema de Cifrado.

Como se describía en este apartado, el proceso consta de varios componentes diferenciados entre sí, debido a su complejidad.

5.4.1. OBTENCIÓN DE CONDICIONES

Una vez la aplicación obtiene la imagen y está lista para ser cifrada, se necesitan las condiciones de cifrado. Para que éste pueda seleccionarlas, se hará un cambio de actividad, pasando a la clase CipherMethodActivity.java. Dicha activity, mostrará la interfaz en la que el usuario seleccionará las condiciones que quiera introducir como parámetros de cifrado.

El almacenamiento de condiciones, se realizará en un objeto de una nueva clase DTO, CondicionesDTO.java. Se ha elegido utilizar esta estructura puesto que resulta mucho más manejable que una estructura de datos simple como un ByteArray o un String. Además de que, pueden existir condiciones nulas, lo cual dificultaría el acceso a los parámetros en los tipos nativos de Java.

Sin embargo los parámetros no se seleccionarán en esta misma clase, sino en las clases que corresponden a tal fin, cada cual asociada a su parámetro, es decir, las condiciones que seleccionará el usuario se almacenarán en el objeto CondicionesDTO en las actividades PasswordActivity.java, WifiIdActivity.java, DateActivity.java y UserActivity.java, que se corresponden al almacenamiento de la contraseña, el SSID del punto de acceso a WI-FI, la fecha y la hora y el usuario, respectivamente.

El funcionamiento de PasswordActivity es simple. La interfaz mostrada cuenta con un campo de texto en el cual el usuario introducirá su contraseña. Si el usuario pulsa sobre el botón “Continuar” se procederá a la validación de la contraseña antes de realizar el almacenamiento en el CondicionesDTO. Si la contraseña es válida, se almacena y mediante un intent se vuelve a CipherMethodActivity, con el fin de que el usuario pueda seguir introduciendo parámetros nuevos o modificar los que ya tiene.

Para que una contraseña sea válida, debe ser mayor de ocho caracteres. Además debe contar con al menos, una mayúscula, una minúscula, un número y un carácter especial del teclado de Android.

El funcionamiento es similar para el caso de DateActivity. El usuario introduce una fecha y una hora mediante un dialog distinto asignado a cada opción. Pulsando el botón de selección de fecha, se mostrará un dialog con un sencillo selector de calendario. Pulsando el de selección de hora, se mostrará el selector de hora por defecto de la versión de Android. La validación en este caso no se realiza del mismo modo, puesto que tanto fecha como hora podrán ir vacías. Únicamente se tendrá en cuenta que cuando la fecha sea anterior a la actual no será necesario almacenarla en el DTO.

WifiIdActivity tiene un funcionamiento más complejo. El usuario puede introducir manualmente el código de la puerta de enlace del punto de acceso a WI-FI en un campo de texto preparado para tal fin. Sin embargo, puesto que esto puede no ser una solución muy

adecuada para el usuario final, se implementará un único botón para identificar la red actual. Es evidente que si el terminal no está conectado a una red WI-FI en el momento que se va a identificar el código, dicha identificación fallará, para eso, cuando el usuario pulse dicho botón se seguirán los pasos descritos en el diseño. En primer lugar, se comprobará si el terminal tiene el servicio WI-FI habilitado. En caso de no ser así, se mostrará un mensaje informativo y se le dará el control al usuario de habilitar el servicio. Como se ha remarcado anteriormente, es importante que sea el usuario el que tome esta decisión. Una vez habilitado el servicio WI-FI, se pasa a obtener la conexión. Si el usuario no está conectado a ninguna red, también se mostrará un mensaje, instándole a conectarse a la red adecuada. Solo cuando el servicio esté activado, y el terminal conectado, se pasará a obtener el SSID del puerto de acceso.

Finalmente, UserActivity, cuenta con un único campo de texto, en el cual el usuario introduce el nombre de los usuarios que serán los destinatarios de la imagen cifrada, una vez completado el proceso. Los usuarios irán separados por comas y será tarea de esta actividad diferenciar a los usuarios para almacenarlos como parámetros en el objeto de CondicionesDTO.

El si una de las condiciones está vacía se inicializará un token de control, distinto para cada parámetro y si la condición está informada (no vacía), se colocará un token distinto antes de la misma. El fin de esto es que a la hora de descifrar las condiciones, se pueda realizar una identificación de cada parámetro de manera rápida, puesto que el total de la cadena de caracteres con las condiciones (sin cifrar) será de tamaño variable.

Una vez obtenidas todas las condiciones que el usuario haya creído oportunas, CipherMethodActivity, las recupera y las enviará mediante un intent como cadena de caracteres, incluyendo los tokens de las condiciones, a la siguiente actividad, preparándolas para el cifrado.

5.4.2. GENERACIÓN DE CLAVE DE CIFRADO

Para el proceso de cifrado en sí, se creará una nueva clase Cifrador.java. En ella se implementará el contenido del subcomponente cifrado de la imagen del subsistema de cifrado. En dicha clase, se deberán encontrar los siguientes métodos:

- El método generarSalt() que generará un salt aleatorio para introducirlo como parámetro en el cifrador.
    - Para ello deberá declararse un objeto de tipo SecureRandom para generar el salt.
    - Posteriormente se realizará una función resumen de ese salt para concatenarlo al propio salt, para verificar la integridad del mismo.
- El método derivarClave(), que se encargará de realizar la derivación de la contraseña.
    - En primer lugar, se generará un número aleatorio que se corresponderá con el número de iteraciones que se va a aplicar al proceso de derivación.
    - Se inicializa un bucle para derivar la contraseña ese número de veces. Recordamos que según el proceso de diseño realizado en el presente documento, el algoritmo de derivación PBKDF2 y con padding PKCS5.
    - El número aleatorio deberá ir adjunto a la contraseña derivada.
- El método cifrar().Que se detallará más adelante.

Sabiendo que la contraseña introducida es suficientemente segura (ya se ha validado en la clase PasswordActivity), debemos aumentar aún más su fortaleza mediante la derivación. Este proceso, como se comentaba con anterioridad, implicará que una misma imagen con una misma contraseña introducida, no sea igual una vez se realice el proceso de cifrado. Esto, hará a la imagen cifrada mucho más difícil de descifrar con métodos no autorizados.

5.4.3. CIFRADOR

En este apartado se detalla el propio proceso de cifrado de la imagen. Como se ha establecido en el apartado de diseño, el cifrador deberá reunir los parámetros necesarios para ejecutarse, por tanto la implementación deberá seguir los siguientes pasos:

- Se declarará un nuevo método cifrar() en la clase Cifrador.java. A dicho método se le pasarán como parámetros:
    - La clave derivada.
    - La imagen previamente tratada.
    - Un número aleatorio “n”.
- El método declarará un objeto de tipo Cipher con los parámetros AES/CBC/PKCS5PADDING en modo cifrado (ENCRYPT). De esta manera, implementaremos el cifrador con el modo indicado determinado anteriormente.
- Se declarará un objeto de tipo SecretKeySpec con la clave derivada.
- Se declarará un vector aleatorio de inicialización, nos referiremos a éste como “IV”
- Se ejecutará el cifrado de los bytes que componen la imagen con la clave SecretKeySpec, el salt y IV un total de n veces, siendo n el numero aleatorio mencionado en el primer punto.
- Al resultado se concatenará el salt, IV y n.

Se siguen los mismos pasos para el cifrado de las condiciones. Una vez cifrados ambos elementos, se concatenan para obtener el resultado final de la imagen cifrada.

El resultado final del cifrado se enviará a la pantalla de confirmación.

5.5. MÓDULO DESCIFRAR

Para implementar el módulo de descifrado, debemos definir por separado varios componentes, del mismo modo que hacíamos con el cifrado.

Sin embargo, en primer lugar, necesitaremos una nueva clase Java que se encargue de reunir todas las tareas del módulo. Descifrador.java será el nombre que reciba.

A continuación, se van a definir, por separado pero siguiendo el orden de ejecución en un proceso de descifrado, los componentes del presente módulo.

5.5.1. COMPROBACIÓN DE CONDICIONES

En primer lugar, sabemos que lo que obtenemos como parámetro de entrada es una imagen correctamente cifrada y con el formato adecuado para su tratamiento.

Por tanto, primero es necesario separar las condiciones del resto de la imagen. Para ello se implementará un método separarCondiciones(). Dicho método Java, realizará un recorrido secuencial a lo largo de los bytes de la imagen cifrada hasta encontrar la marca que delimita la parte del fichero que corresponde a las condiciones de cifrado. Una vez encontrada la marca, separa ambas partes para su posterior tratamiento.

Es entonces cuando la aplicación trabajará con las condiciones de cifrado obtenidas. Antes de poder hacer algo con ellas, es necesario descifrarlas, por tanto, se realizará una llamada al componente de descifrado. Una vez se hayan descifrado correctamente, se asume que las condiciones están listas para su comprobación.

Antes de comprobar cada una de ellas, se comprueba que las condiciones hayan sido introducidas por el usuario cifrante, es decir, si están vacías o no. Evidentemente no se podrán comprobar condiciones vacías. Las comprobaciones se realizan fácilmente gracias a los token de control utilizados, que permitirán separas las condiciones unas de otras.

La primera condición que se comprueba será la contraseña. Existirá un parámetro que indique si el usuario introdujo una contraseña secreta concreta o si por el contrario la imagen se ha cifrado con la contraseña por defecto de la aplicación. Si el parámetro indica que la contraseña es secreta, se requerirá al usuario que intenta descifrar la imagen que introduzca la misma contraseña. En caso de que no sea igual, se lanzará un mensaje de error.

Posteriormente, se comprueba la fecha. Como se ha mencionado anteriormente, si la comprobación de la fecha que se realiza en este punto tomara como referencia la fecha y la hora del sistema operativo Android, el usuario podría saltar esta condición de manera ilícita con el sencillo hecho de adelantar la hora de su terminal. Por esto, se realiza una comprobación de la hora con un servidor externo. Para la llamada a este servicio, se crea una nueva clase utilitaria denominada GetDateTask().

GetDateTask ejecuta una tarea asíncrona (AsyncTask) para la llamada al servidor. Una vez conectado el terminal, se lanza una petición. El servidor envía como respuesta una cadena de caracteres con la hora actual en dicho servidor. Posteriormente se da formato a la fecha.

Para comparar las fechas, se crea una nueva clase, DateComparator.java. Dicha clase posee un sencillo método que convierte una cadena de caracteres con una fecha a formato Date, stringToDate(). Esto se realiza para poder comparar dos fechas de manera efectiva con el método comparar(), que determina qué fecha es mayor.

Así pues, podemos determinar si la fecha actual es mayor que la fecha obtenida de las condiciones de cifrado y por tanto, si se permitirá descifrar la imagen.

Para analizar la condición de WI-FI, si existe, se debe, antes que nada, comprobar que la conexión WI-FI del terminal está habilitada. Del mismo modo que en el cifrado, se solicita permiso al usuario para habilitar dicho servicio en caso de que no esté ya conectado a una red. Así, la comprobación es sencilla, si la condición WI-FI de cifrado obtenida coincide con el SSID de la red a la que el dispositivo está conectado actualmente, se permitirá el

descifrado de la imagen, de cualquier otro modo, no se permitirá, lanzando un mensaje de error y cancelando el proceso.

Finalmente, debemos comprobar la condición correspondiente al usuario de Topoos. Si la condición está informada, debemos obtener nuestro propio nombre de usuario de la plataforma. Para ello necesitamos la clase GetUserTask.java. Dicha clase, no consiste en una tarea asíncrona como en el caso de la fecha, sino que crea un hilo para lanzar una petición al servidor de Topoos y así obtener los datos necesarios. Sin embargo, dada su similitud, se ha decidido utilizar esa nomenclatura. GetUserTask extiende de la clase Java Runnable, pues es capaz de ejecutarse como proceso ligero. La clase recupera el token de autenticación de la aplicación. Cuando se ejecuta, utilizando el API de Topoos, se envía una petición para obtener el usuario autenticado en la aplicación y después el nombre de usuario del mismo. Una vez hecho esto, se debe hacer uso de un método *handler* en Descifrador.java para obtener el parámetro y poder comprobar si coincide con el nombre de usuario que introdujo el usuario cifrante.

Una vez comprobadas todas las condiciones, si se cumplen, el módulo las ignorará desde este punto, puesto que comenzará a trabajar con el grueso de la imagen.

5.5.2. GENERACIÓN DE CLAVE DE DESCIFRADO

Una vez tengamos la contraseña (ya sea privada o la que la aplicación asigna en su defecto), podemos comenzar con la generación de clave de descifrado. Hay que tener en cuenta que para el caso del descifrado de condiciones, la contraseña siempre será la contraseña por defecto.

Primeramente, debemos separar el *salt* y el número de iteraciones que se han utilizado para la derivación de contraseña para cada caso (condiciones e imagen). Dado que se tratan de datos de tamaño fijo, sencillamente debemos obtenerlos del final del flujo de bytes obtenidos de los datos de entrada.

Con todos los datos necesarios, sencillamente debemos seguir los pasos definidos en la generación de contraseña del módulo de cifrado. De este modo, si la contraseña fuese incorrecta, la imagen o las condiciones que se fueran a descifrar después tampoco serían correctas, y se produciría un error en el proceso de descifrado.

5.5.3. DESCIFRADOR

El proceso de descifrado es muy similar al de cifrado desde el punto de vista de la implementación, ya que se utiliza el mismo algoritmo, solo que en modo de descifrado.

Así, los pasos a seguir serán los siguientes:

- Se declarará un nuevo método descifrar() en la clase Descifrador.java. A dicho método se le pasarán como parámetros:
    - La clave de descifrado derivada.
    - Los datos cifrados, ya sea la imagen o las condiciones.

    - El número de iteraciones “n” recuperado (viene concatenado a los datos).
    - El vector IV de inicialización.
- El método declarará un objeto de tipo Cipher con los parámetros AES/CBC/PKCS5PADDING, pero esta vez, en modo descifrado (DECRYPT).
- Se declarará un objeto de tipo SecretKeySpec con la clave derivada.
- Se ejecutará el descifrado de los bytes que componen la imagen o las condiciones con la clave SecretKeySpec, el *salt* y IV un total de n veces, siendo n el numero aleatorio mencionado en el primer punto.

Una vez obtenidos los datos descifrados, en caso de que se traten de las condiciones, se realizará el tratamiento como se ha descrito en la comprobación de condiciones. En caso de tratarse de la imagen, se envía al módulo de pantalla de confirmación.

5.6. MÓDULO DE PANTALLA DE CONFIRMACIÓN

En el presente módulo, tanto como si la acción que realiza el usuario de la aplicación es de cifrado como si es de descifrado, se deben implementar los dos subsistemas que requieren de la creación de una imagen a partir de los datos obtenidos. Dichos subsistemas, que se definieron en el diseño del sistema de información, son el envío y el almacenamiento de las imágenes.

Además, si el usuario marcó la opción para eliminar la imagen original, se deberá eliminar en este momento.

Puesto que para ambos casos se requerirá dicha creación, se implementará una nueva clase Java, ImageSaver.java destinada a realizar esa tarea.

ImageSaver contará con un método save() al que se le pasarán como parámetro los datos necesarios para renderizar los datos. Los datos serán:

- El grueso de bytes que compondrán la imagen
- La codificación (JPG, PNG…)

El método crea un objeto de tipo Bitmap en el que mediante la acción *compress* introduce los datos. Es decir, se comprimirán los bytes de la imagen con la codificación especificada.

Dependiendo de la acción que el usuario haya realizado (cifrado o descifrado) se cargará una interfaz u otra que muestre el mensaje de confirmación que corresponda. Para ello, nos valdremos de la clase ConfirmationActivity.java.

En caso de que el usuario haya cifrado la imagen, se mostrará el mensaje de finalización del proceso, pero no la imagen cifrada, puesto que no tendrá sentido verla. Asimismo, se le dará al usuario la opción de almacenar o enviar la imagen cifrada.

En caso contrario, de que el usuario haya descifrado la imagen, también se mostrará el mensaje de confirmación de que el proceso de descifrado ha concluido y en esta ocasión sí se mostrará la imagen descifrada (mediante un elemento de interfaz de tipo ImageView), dado que no se le dará al usuario la opción de enviarla, tal y como se especifica en el diseño del sistema de información.

Para cada una de las opciones seleccionables por el usuario en la correspondiente interfaz, se van a definir dichas acciones en los siguientes apartados.

5.6.1. ENVÍO DE IMÁGENES

Únicamente si el usuario acaba de terminar de cifrar una imagen, se le dará la opción de acceder al envío de imágenes cifradas por aplicaciones de terceros.

ConfirmationActivity contará con un método enviar() que mediante un intent abrirá un selector de aplicaciones con las que el usuario podrá enviar la imagen. A dicho intent se le deberá inicializar con Intent.ACTION_SEND, de otro modo no se podrá enviar ninguna imagen.

5.6.2. ALMACENAMIENTO DE IMÁGENES

Existen dos casos en los que el usuario puede llegar a este componente de implementación. La única diferencia se detallaba en el apartado del diseño. Si el usuario acaba de cifrar una imagen y en la clase ConfirmationActivity se ve reflejado esto mismo, el usuario almacenará las imágenes en una carpeta que creará la aplicación en el momento de su instalación. Por el contrario, si el usuario ha descifrado, la carpeta donde se almacenaran las imágenes descifradas será una carpeta a la que la galería de Android no tendrá acceso. Esto se consigue con la creación de un fichero vacío llamado ".nomedia" en el interior de la carpeta que se pretende proteger.

Los motivos de esta decisión vienen detallados en el diseño. Centrándonos en la implementación, salvo esa pequeña diferencia, que no es más que la ruta de almacenamiento, el método almacenar() de la clase que se mencionaba en el párrafo anterior, será el mismo para ambos casos.

Almacenar(), primero, decide, en primer lugar la ruta de almacenamiento. Una vez declarada, se creará un objeto File con el path de esa ruta.

Posteriormente, se consulta la fecha y la hora (en esta ocasión la del sistema operativo ya que no compromete la seguridad y se trata de un mero trámite utilitario). La fecha y la hora, servirán para guardar la imagen con un nombre único, de manera que no se repita ningún nombre de fichero. Aun así, se realiza una comprobación por si el nombre con el que se almacenará la imagen en la ruta ya existe previamente. En tal caso, el fichero se sobre-escribirá. Hay que prestar especial cuidado a la extensión del fichero a la hora de almacenarlo.

Finalmente se declarará un objeto FileOutputStream al que se le añadirá la imagen en sí mediante el método write(). Una vez finalizado este proceso, es importante para la memoria del dispositivo realizar el close() del objeto. De esta forma, obtendremos l imagen guardada en la memoria del terminal.

5.7. OTROS MÓDULOS

Existen otros aspectos, distintos a los que ya se han definido que es necesario implementar. En el desarrollo de Image Cipher, se han definido interfaces con errores. Sin embargo hasta ahora no se ha hecho mención de cómo se muestran dichos errores.

La respuesta es relativamente sencilla, a lo largo de la implementación del proyecto, se deberá prestar atención a aquellos errores que puedan producirse (error al cifrar, la imagen es demasiado pequeña, etc.) Dichos errores controlados, se mostrarán al usuario de la siguiente forma:

- En caso de producción de error, en primer lugar se debe capturar la excepción utilizando un catch().
- En función del tipo de excepción, y para cada caso, se definirá un mensaje concreto y suficientemente descriptivo para mostrar al usuario.
- Se inicializará la interfaz de error con el mensaje de la excepción escrito en ella.
- Se bloqueará el proceso que se esté ejecutando (cifrado, apertura de imágenes, almacenamiento, etc.).
- Se forzará el botón Back de Android para que cuando se pulse una única vez se salga de la aplicación.

Para ello no se utilizará una clase externa que maneje los errores, sino que dentro del propio catch() con escasas líneas de código se podrá manejar la gestión de errores controlados.

Otro aspecto a tener en cuenta es el apartado de novedades y versión de la aplicación. Se trata simplemente de una interfaz en la que el usuario no puede hacer más que volver a la pantalla anterior, que es la principal (WelcomeActivity). Por ello, se deberá implementar un botón que sencillamente cargue esta interfaz con los datos de la versión instalada estáticamente definidos en ella, es decir, los textos no van a cambiar en ningún caso, salvo la actualización de la aplicación.

5.8. ESTRUCTURA DEL PROYECTO

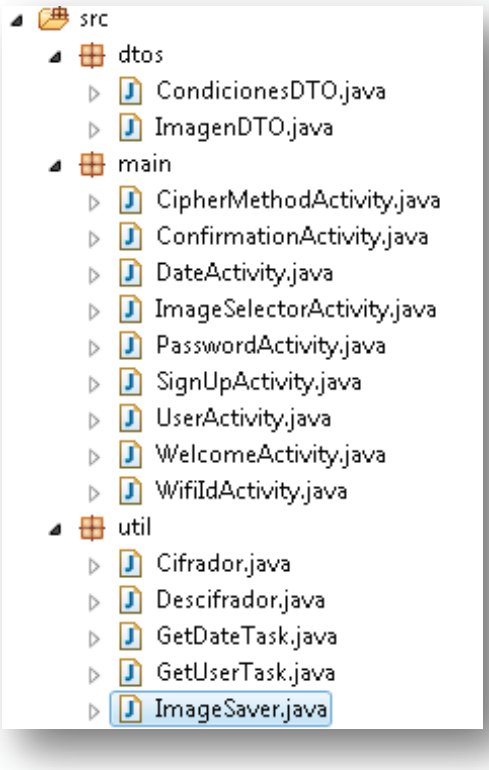


ILUSTRACIÓN 57: APLICACIÓN IMPLEMENTADA

6. EVALUACIÓN Y RESULTADOS

El desarrollo de una aplicación nunca acaba con la implementación. Se debe hacer siempre una batería de pruebas suficientes como para asegurar que la aplicación funciona. Asimismo, aunque no se vaya a incluir en el presente documento, se debe cumplir un compromiso con el mantenimiento de las mismas, por futuros errores que pueden surgir desde la publicación de la aplicación.

Por tanto, a continuación se van a realizar varias pruebas diferenciadas en grandes grupos: funcionalidad, rendimiento, integridad y seguridad.

Para realizar las pruebas, se han establecido una serie de condiciones o *test* que la aplicación deberá cumplir de forma que se evalúe correctamente. Dado el número de dichos test, y con el fin de que queden bien identificados, se opta por utilizar un formato tabular para la definición de los mismos.

La tabla de identificación de los test es la siguiente:

TX - 00			
Descripción:			
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 101: EJEMPLO DE TEST

Los campos de la tabla son los siguientes:

- TX - 00: Especifica la naturaleza del test, así como su número:
  - La X toma los valores: F, I, S. Funcionalidad, Integridad y Seguridad respectivamente. Estos conceptos se definen en los sub-apartados siguientes.
- Descripción: La descripción del test.
- Importancia: Especifica la importancia del test. Únicamente tiene fin orientativo.
- Resultado: Si se cumple o no el test de evaluación.

La tabla de identificación de test de prueba no se aplicará a la evaluación de rendimiento. Esto es porque se utilizará un análisis estadístico en su lugar.

6.2. FUNCIONALIDAD

Las pruebas de funcionalidad ayudan a asegurar que la aplicación funciona en todos sus aspectos, es decir, que cumple todas sus funcionalidades. Se deberán realizar comprobando el mayor número de requisitos posibles.

Hay que remarcar que la evaluación de funcionalidad únicamente se centra en si los requisitos se cumplen o no. El cómo se realicen desde otros puntos de vista corresponderán a las pruebas de rendimiento e integridad.

6.2.1. LOG-IN

TF - 01			
Descripción:	La aplicación despliega correctamente.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 102: TF-01

TF - 02			
Descripción:	Si no existe autenticación previa, se redirige al usuario al formulario de autenticación.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 103: TF-02

TF – 03			
Descripción:	La autenticación se lleva a cabo únicamente en el caso de que el usuario ya esté registrado.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 104: TF-03

TF – 04			
Descripción:	El usuario puede acceder al formulario de registro a través del enlace en el formulario de autenticación.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 105: TF-04

TF – 05			
Descripción:	Una vez registrado, el usuario puede autenticarse.		
Importancia:	Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 106: TF-05

TF – 06			
Descripción:	Una vez autenticado, se redirige al usuario a la pantalla de bienvenida.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 107: TF-06

6.2.2. APERTURA DE IMÁGENES

TF – 07			
Descripción:	El usuario podrá acceder a la pantalla de selección de imágenes mediante los botones en la pantalla principal.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 108: TF-07

TF – 08			
Descripción:	El usuario podrá acceder a la galería mediante el botón destinado a tal fin.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 109: TF-08

TF - 09			
Descripción:	La imagen que el usuario selecciona en la galería se muestra en la pantalla de selección de imagen.		
Importancia:	☐Alta	☐Media	☑Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 110: TF-09

6.2.3. CIFRADO DE UNA IMAGEN

TF – 10			
Descripción:	El usuario puede acceder al módulo de cifrado de una imagen a través del botón en la pantalla principal.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 111: TF-10

TF – 11			
Descripción:	El usuario puede seleccionar cualquier botón para añadir condiciones para acceder a cada una de ellas.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 112: TF-11

TF – 12			
Descripción:	El usuario puede introducir una password en la ventana de password.		
Importancia:	Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 113: TF-12

TF – 13			
Descripción:	El usuario puede introducir un SSID en la ventana de WI-FI.		
Importancia:	Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 114: TF-13

TF – 14			
Descripción:	El usuario puede establecer el SSID del punto de acceso actual en la ventana de WI-FI si está conectado a una red.		
Importancia:	☐Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 115: TF-14

TF – 15			
Descripción:	El usuario puede introducir una fecha y una hora como parámetro en la ventana de Fecha / Hora.		
Importancia:	☐Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 116: TF-15

TF - 16			
Descripción:	El usuario puede introducir el nombre de otros usuarios en la ventana de usuario.		
Importancia:	Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 117: TF-16

TF – 17			
Descripción:	El usuario podrá eliminar la imagen original.		
Importancia:	☐Alta	☐Media	☑Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 118: TF-17

TF – 18			
Descripción:	Cuando el usuario confirme la acción de cifrar, la pantalla de proceso de muestra correctamente.		
Importancia:	☐Alta	☐Media	☑Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 119: TF-18

TF – 19			
Descripción:	Cuando el proceso de cifrado finaliza se muestra la pantalla de confirmación.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 120: TF-19

TF – 20			
Descripción:	La imagen resultante del proceso de cifrado es ininteligible para el sistema.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 121: TF-20

6.2.4. DESCIFRADO DE UNA IMAGEN

TF – 21			
Descripción:	El usuario podrá acceder al módulo de descifrado de imágenes pulsando en el botón destinado a tal fin en la pantalla principal.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 122: TF-21

TF – 22			
Descripción:	Si se requiere una contraseña, la ventana correspondiente se despliega correctamente.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 123: TF-22

TF – 23			
Descripción:	Si una de las condiciones de descifrado no se cumple de despliega un mensaje informando al usuario.		
Importancia:	☐Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 124: TF-23

TF – 24			
Descripción:	Cuando comienza el descifrado de la imagen, se despliega la pantalla de proceso de descifrado correctamente.		
Importancia:	☐Alta	☐Media	☑Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 125: TF-24

TF – 25			
Descripción:	Cuando el proceso de descifrado concluye, se muestra correctamente la pantalla de confirmación.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 126: TF-25

TF – 26			
Descripción:	La imagen resultante del proceso de descifrado es una imagen coherente y válida.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 127: TF-26

TF – 27			
Descripción:	La imagen resultante del proceso de descifrar se mostrará en la pantalla de confirmación.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 128: TF-27

6.2.5. COMPARTICIÓN DE CÓDIGO CON APLICACIONES DE TERCEROS

TF – 28			
Descripción:	Cuando el usuario pulsa sobre el botón “Enviar” se despliega el selector de aplicaciones compatibles de forma correcta.		
Importancia:	☐Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 129. TF-28

TF – 29			
Descripción:	Se envía la imagen a la aplicación seleccionada de forma correcta.		
Importancia:	☐Alta	☑Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 130: TF-29

6.2.6. ALMACENAMIENTO

TF – 30			
Descripción:	Cuando el usuario, después de cifrar, pulsa sobre el botón de “Almacenar” el usuario podrá especificar la ruta donde quiere guardar la imagen cifrada.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 131: TF-30

TF - 31			
Descripción:	La imagen cifrada se almacena en la ruta seleccionada y de forma correcta.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 132: TF-31

TF – 32			
Descripción:	Cuando el usuario, después de descifrar, pulsa sobre el botón "Almacenar" la imagen descifrada se almacenará correctamente en la carpeta oculta.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 133: TF-32

TF – 33			
Descripción:	La carpeta oculta no será accesible por la galería de Android.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 134: TF-33

6.2.7. NOVEDADES Y VERSIÓN

TF – 34			
Descripción:	Se desplegará la pantalla de novedades y versión correctamente cuando el usuario pulse sobre el enlace de la pantalla principal.		
Importancia:	☐Alta	☐Media	☑Baja
Resultado:	☑Se cumple	☐No se cumple	

TABLA 135: TF-34

6.3. RENDIMIENTO

Una vez completada la evaluación funcional de la aplicación implementada se deberá evaluar el rendimiento de la misma. Es importante que el rendimiento no sea muy elevado, dado que existirán dispositivos de rendimiento muy limitado.

Mediremos el rendimiento con la velocidad con la que la aplicación será capaz de realizar las funcionalidades con datos de mayor tamaño. Existen funcionalidades que no se han considerado evaluar, tales como la apertura de imágenes, guardado o envío de las mismas o errores de maquetación o interfaces. Esto se justifica porque no tienes una importante relación con el tiempo de ejecución o el tamaño de los datos.

Sin embargo el rendimiento no se mide bien con test similares a los test de funcionalidad. Para evaluar de forma coherente el rendimiento de la aplicación necesitaremos dos

terminales para realizar pruebas, que no son más que, como se indicaba, varias ejecuciones de la aplicación. De esa forma y mediante una estadística encontraremos que las pruebas son o no satisfactorias.

Los terminales que se van a utilizar, son los mencionados en el apartado de entorno operacional. Se definieron un dispositivo más antiguo, un Samsung Galaxy Ace, al cual nos referiremos como Terminal 1, también se utilizará un smartphone más potente, Samsung Galaxy S III- Mini, en adelante Terminal 2.

A continuación, se evaluarán las funcionalidades de cifrado y descifrado de imágenes.

6.3.1. CIFRADO DE UNA IMAGEN

La primera funcionalidad que se analiza es el cifrado de una imagen. Los tiempos que se muestran en la tabla de resultado son los que se han medido mediante ejecuciones consecutivas de la aplicación, desde el momento en el que se realiza el cifrado en sí, con todas las condiciones informadas y previamente cifradas, hasta el momento en el que se finaliza el proceso, antes de cargar la interfaz de la pantalla de confirmación.

Esto se debe a que lo que realmente se quiere medir es el tiempo de cifrado en proporción con el tamaño de imagen. Las condiciones, también son cifradas pero su tamaño es reducido y limitado, por lo que no se ha considerado útil para el informe estadístico.

Los resultados obtenidos en las pruebas son los siguientes:

Resolución de la Imagen	Tamaño aproximado	Tiempo Terminal 1	Tiempo Terminal 2
512x512	36KB	1.825s	0.53s
683x683	52KB	1.952s	0.64s
800x600	76KB	2.06s	0.8s
1024x1024	1MB	2.1s	0.93s
2048x2048	2MB	2.17s	1.18s

TABLA 136: TIEMPOS DE CIFRADO

Los cuales resultan en el gráfico que se muestra a continuación:

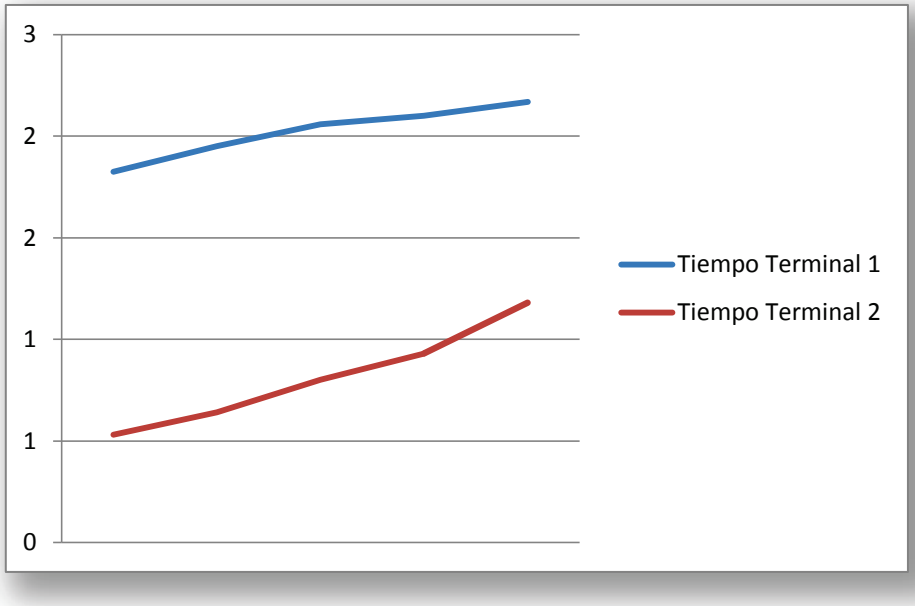


ILUSTRACIÓN 58: TIEMPOS DE CIFRADO

En los resultados vemos que el Terminal 1 (Samsung Galaxy Ace) tiene tiempos de ejecución bastante superiores, esto se debe a la limitación de recursos hardware, algo menos notable en el Terminal 2 (Samsung Gallaxy S III Mini).

Los tiempos de ejecución no crecen de forma lineal, ya que las muestras no siguen tal relación entre sí. Tampoco lo hace de exponencial lo que indica la ausencia de errores en el proceso.

Los puntos de inflexión pueden deberse a ciertos factores, muy probablemente relacionados con la memoria disponible en el dispositivo en ese momento.

De cualquier forma, el objetivo primero de este análisis era cumplir el requisito de usuario RUR-08, en el que se especifica que "*El tiempo de cifrado o descifrado, en el peor de los casos, no excederá de 1 minuto*".

6.3.2. DESCIFRADO DE UNA IMAGEN

También se realiza un pequeño estudio estadístico sobe la funcionalidad de descifrado de una imagen. En estas pruebas de rendimiento, y por motivos del diseño e implementación, no es posible separar la lectura y análisis de condiciones de descifrado del propio proceso. Se ha tenido, por tanto, especial cuidado en cumplir todas las condiciones de descifrado en el momento de realización de las pruebas.

La resolución de las imágenes mostradas en la tabla que hay a continuación son aproximadas, y se corresponden a las imágenes que se obtienen como resultado de la prueba de cifrado realizada en el apartado anterior.

Los resultados obtenidos son los siguientes:

Resolución de la Imagen	Tamaño aproximado	Tiempo Terminal 1	Tiempo Terminal 2
512x512	36KB	3s	1,86s
683x683	52KB	4s	2,07s
800x600	76KB	3,91s	2,46s
1024x1024	1MB	4,02s	2,74s
2048x2048	2MB	4,12s	3,19s

TABLA 137: TIEMPOS DE DESCIFRADO

Con los datos de la tabla anterior, obtenemos el siguiente gráfico:

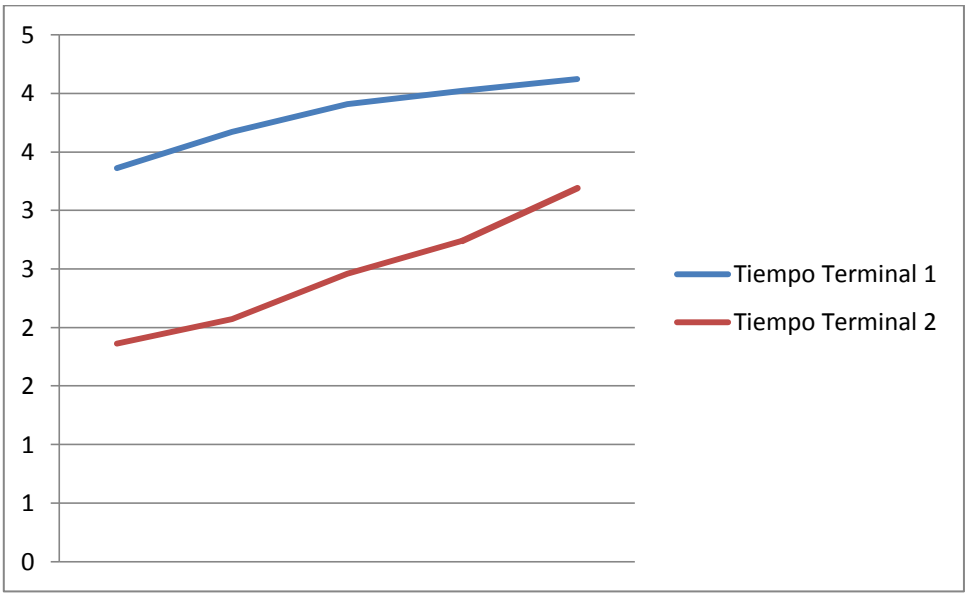


ILUSTRACIÓN 59: TIEMPOS DE DESCIFRADO

En el gráfico vemos que en comparación con el proceso de cifrado, los tiempos aumentan moderadamente. Esto se debe a lo ya explicado, es decir, no se han podido separar la comprobación y descifrado de las condiciones adjuntas en el fichero de imagen.

Aun así, vemos una gráfica similar al apartado anterior. Esto se produce porque las condiciones descifradas son las mismas para todas las muestras y el algoritmo de descifrado es el mismo que el de cifrado, pero en sentido inverso.

Vemos que de la misma forma que en el cifrado se cumple ampliamente lo especificado en el requisito de usuario RUR-08.

6.4. INTEGRIDAD

Las pruebas de integridad serán las que lleven a la aplicación a situaciones límite, serán las pruebas más complicadas en cuanto a rendimiento o situaciones anormales.

6.4.1. RECUPERAR LA IMAGEN CIFRADA DE FORMA ILÍCITA

Para este primer bloque de test de integridad se tratará de comprometer la imagen cifrada con el incumplimiento de las condiciones, de forma que ataquen contra la propia integridad de la imagen y de la aplicación. Al contrario que para el caso de los test de funcionalidad, se dará válido el análisis de integridad si no se cumple ninguno de los test que se definen a continuación:

TI - 01			
Descripción:	Descifrar la imagen incumpliendo la condición de password.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 138: TI-01

TI - 02			
Descripción:	Descifrar la imagen alterando el salt de la imagen cifrada.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 139: TI-02

TI - 03			
Descripción:	Descifrar la imagen incumpliendo la condición de WI-FI ID.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 140: TI-03

TI - 04			
Descripción:	Descifrar la imagen incumpliendo la Fecha / Hora.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 141: TI-04

TI - 05			
Descripción:	Descifrar la imagen cambiando la fecha / hora del dispositivo.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 142: TI-05

TI - 06			
Descripción:	Descifrar la imagen incumpliendo la condición de usuario.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 143: TI-06

TI - 07			
Descripción:	Descifrar la imagen sin conexión a internet.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 144: TI-07

De esta forma, vemos que cumplimos los requisitos de restricción RUR-09.

6.4.2. OTROS TEST DE INTEGRIDAD

Con el fin de cubrir el máximo número de requisitos de restricción en las pruebas de integridad de Image Cipher, se ha optado por añadir los siguientes test:

TI – 08			
Descripción:	Descifrar la imagen accediendo a la aplicación sin realizar la autenticación.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 145: TI-08

TI - 09			
Descripción:	Acceder a imágenes descifradas con la galería de Android.		
Importancia:	☐Alta	☑Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 146: TI-09

TI - 10			
Descripción:	Autenticarse con una contraseña o nombre de usuario no válido.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 147: TI-10

TI - 11			
Descripción:	Introducir una contraseña no válida.		
Importancia:	☑Alta	☐Media	☐Baja
Resultado:	☐Se cumple	☑No se cumple	

TABLA 148: TI-11

Así pues, comprobamos los requisitos de restricción restantes, a excepción de los que tienen que ver con la interfaz, ya tenidos en cuenta en la fase de implementación.

6.5. SEGURIDAD

A continuación se va a analizar la fortaleza del algoritmo utilizado para el desarrollo de Image Cipher.

Como se ha venido explicando a lo largo del presente documento, el algoritmo utilizado es el cifrador de bloque AES con un tamaño de clave de 256 bits. Actualmente se trata de un algoritmo de cifrado que no se ha roto todavía, es decir, no es vulnerable en ese sentido. Por tanto, el único punto débil, matemáticamente hablando sería mediante un ataque por fuerza bruta.

Los ataques por fuerza bruta (o *brute force*), consisten en probar todas las combinaciones posibles para lograr descifrar un mensaje, en este caso, la imagen previamente cifrada. Esto es computacionalmente muy costoso, más teniendo en cuenta que se han utilizado medidas de seguridad adicionales para dificultar precisamente este tipo de ataques. Esas medidas son el *salt* y la derivación de clave. Además el propio algoritmo cuenta con la fortaleza del vector de inicialización aleatorio, que también protege los datos cifrados ante este tipo de acciones maliciosas.

Sin embargo, en un futuro, puede que próximo, sea computacionalmente posible romper el cifrado que se ha implementado en Image Cipher. Puesto que el tamaño de clave es de 256 bits, para conseguir la clave adecuada, se necesitarían un total de $1.1\ x\ 10^{77}$ combinaciones a cubrir. No hay que olvidar, que la clave no se introduce tal cual la introduce el usuario, sino que pasa por el proceso de derivación. Si un dispositivo, equipo de sobremesa o *cluster* tardara solamente 1s. en generar la clave, derivarla y comprobar el resultado, necesitaríamos un total de $3{,}47\ x\ 10^{84}$ años. Podemos concluir que, de momento, el algoritmo AES, y por tanto, también la aplicación son suficientemente seguros.

7. CONCLUSIONES

Finalizado el proyecto, y a modo de conclusión, únicamente queda comprobar cuantos de los objetivos planteados al comienzo del mismo se han cumplido y si se ha dejado alguno parcialmente incompleto.

Asimismo, se deberían plantear cuestiones que resulten, al menos, interesantes para realizar como trabajos futuros, en caso de que se pretenda ampliar el alcance del sistema.

7.2. OBJETIVOS ALCANZADOS

En el apartado de objetivos iniciales para el proyecto, se planteaba el desarrollo de un cifrador de imágenes multicondición, bajo el nombre de Image Cipher. Estos son los objetivos cumplidos:

- Aplicación de cifrado de imágenes.
- Integración con el sistema operativo móvil Android.
- Cifrado simple de imágenes (con contraseña).
- Derivación de contraseña.
- Salt.
- Cifrado de imágenes sin necesidad de contraseña.
- Condición de hora y fecha.
- Comprobación de hora y fecha mediante una fuente externa.
- Condición de WI-FI SSID.
- Posibilidad de activar el servicio de WI-FI.
- Identificación del WI-FI SSID actual.
- Integración con la plataforma Topoos.
- Gestión básica de usuarios en Topoos (registro y autenticación).
- Identificación del usuario actual en Topoos.
- Almacenamiento de imágenes
- Almacenamiento protegido de imágenes (no aparecen en galería).
- Envío de imágenes a aplicaciones de terceros.
- Visualización de imágenes seleccionadas y descifradas.
- Eliminación de imágenes originales en el momento del cifrado.
- Gestión de errores controlados.
- Consulta de novedades de la aplicación.
- Interfaz simple e intuitiva.
- Evaluación del rendimiento.
- Evaluación de seguridad.

En vista de los componentes de la lista anterior, podemos concluir que se han cumplido los objetivos planteados al inicio del proyecto.

7.3. TRABAJOS FUTUROS

Aun habiéndose cumplido los objetivos planteados inicialmente, se ha sido consciente en todo momento de que cualquier proyecto de software es mejorable, y en ocasiones, hasta queda obsoleto con rapidez si no se actualiza.

Por ello se ha optado por proponer ciertos puntos a tener en cuenta en desarrollos futuros a modo de ideas o consejos.

En primer lugar, podría ampliarse en gran medida la utilidad de la aplicación si se implementase la integración de un servicio de geolocalización. Podría añadirse una nueva condición de cifrado "Localización" que únicamente permitiera descifrar una imagen si el usuario se encuentra a una determinada distancia del punto donde se cifró la imagen, o incluso, que en el momento en el que se cifrara la imagen y se añadiera esta condición, pudiera seleccionarse una localización concreta mediante un mapa. Recientemente, la plataforma Topoos ha añadido un servicio de alta y búsqueda de puntos de interés (PoI, Points of Interest) que bien puede servir para esta futura funcionalidad.

Sería muy útil la incorporación de varios idiomas a la aplicación. La aplicación podría buscar el idioma del terminal y adaptarse a las necesidades del usuario en este aspecto. Asimismo, sería muy útil que la hora no se limitara al horario peninsular español.

Un aspecto a tener en cuenta, sería la capacidad de almacenamiento online de las imágenes cifradas y descifradas. Esto supondría un avance en aspectos de seguridad ya que cualquier explorador de ficheros para Android puede acceder a las carpetas "seguras" de la aplicación, y por tanto abrir las imágenes almacenadas en el terminal. Además facilitaría la posibilidad de compartir las imágenes sin necesidad de aplicaciones de terceros. Actualmente, este servicio puede llevarse a cabo a través de la aplicación externa Dropbox, entre otras.

También se es consciente de la mejora de usabilidad que supondría la posibilidad de buscar usuarios de Topoos dentro de la aplicación, puesto que no hay ningún servicio implementado en la aplicación actual que cumpla estos requisitos.

Asimismo, la posibilidad de poder autenticarse mediante Social Login (mediante cuentas de redes sociales como Facebook o Twitter vinculadas) supondría un aumento en la comodidad del usuario al acceder por primera vez a la Image Cipher.

Finalmente, y aunque ha sido muy restrictivo para contemplarlo en el momento inicial del proyecto, sería muy interesante, integrar la aplicación actual para plataformas iOS o iPad.

8. REFERENCIAS

[1]: Definición de cifrado: Real Academia de la Lengua Española:

[2]: Encryption (Wikipedia): Kahn, David, *The Codebreakers - The Story of Secret Writing* (ISBN 0-684-83130-9) (1967)

[3]: Criptografía: The CrypTool Team, *Overview and Applications of Cryptology* —Julio de 2008.

[4]: Ejemplo de cifrado césar: Braingle

http://www.braingle.com/brainteasers/codes/caesar.php

[5]: Concepto de clave: Real Academia de la Lengua Española.

[6]: Discos de Alberti: Morelli, Ralph, *Cryptography Trinity College Department of Computer Science*, 2012.

[7]: Alan Turing: ETSISI Universidad Politécnica de Madrid, 2010.

[8]: Computador Colossus: Copelan, Jack, *Colossus The First Large Scale Electronic Computer*, 21 de Octubre de 2012.

[9]: Cifradores de Bloque y de Flujo: Muñoz Dávila, Lucía , *Investigación Criptográfica*, Instituto Tecnológico de Apizaco. 25 de Septiembre de 2012.

[10]: Smartphone: Phone Scoop. 2011.

[11]: Evolución de los Smartphones: Yousuf*, History Review of the Smartphones Over 20 Years*, TechInfo2. 2013.

[12]: Android 4.4: Google Android Kit Kat.

http://www.android.com/versions/kit-kat-4-4/

[13]: Google Nexus 5 y Iphone 5s, epsecificaciones técnicas: Google Inc., Apple Inc. 2013.

http://www.google.es/nexus/5/

http://www.apple.com/es/iphone-5s/specs/

[14]: Breve historia de Android: Staff, Verge, *Android: A visual history.* 7 de Diciembre de 2011.

[15]: Android alcanza los 9B de apps: Ward, Brad, *Google: 900 million Android activations, 48 billion app installs*, Android Authority. 15 de Mayo de 2013.

[16]: Apple iOS: Apple Inc. España

[17]: Versiones de Apple iOS: Staff, Verge, *iOS: A visual history*, The Verge. 16 de Septiembre de 2013.

[18]: iCloud: Apple Inc., Apple Press Info. 6 de Junio de 2011.

[19]: Siri: Martínez, Javier *Siri en español. Guía de instrucciones y funcionalidades.* iPhone4Spain.com. 17 de Septiembre de 2012.

[20]: iOS 7: Apple Inc, iOS 7.

http://www.apple.com/es/ios/

[21]: X Code: Apple Inc., *XCode, The Complete Toolset for Building Great Apps*, Apple Delevoper

[22]: Hola Mundo en iOS: Rzornoza, *The Init Developers.* 19 de Mayo de 2011.

[23]: Walled Garden: PC Mag Encyclopedia, Personal Computer Magazine, 2011.

[24]: Imagen, definición: Real Academia de la Lengua Española.

[25]: Formatos de imágenes: Instituto Superior de Formación y Recursos en Red para el Profesorado. Ministerio de Educación, Política Social y Deporte. 2008.

[26]: El formato PNG: Roelofs, Greg, *Portable Network Graphics*, The PNG Home Site, 2013.

[27]: El Algoritmo DES: *The National Institute of Standards and Technology (NIST) is an agency of the U.S. Department of Commerce*. Information Tecnology Laboratory.

[28]: El Algoritmo AES: U.S. Government Federal Information Processing Standards *Publication 197 November 26, 2001 Announcing the ADVANCED ENCRYPTION STANDARD (AES).*

[29]: Servicios de Topoos: *Lista de Servicios*, Topoos Beta. 2011.

http://www.topoos.com/servicios

[30]: Ley 32/2003, de 3 de noviembre, General de Telecomunicaciones: Noticias Jurídicas.

http://noticias.juridicas.com/base_datos/Admin/l32-2003.html

[31]: Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal:

http://noticias.juridicas.com/base_datos/Admin/lo15-1999.html

[32]: Comparativa de ventas, según el sistema operativo móvil: Arthur, Charles , *Tablets will challenge PC sales by 2017 as Android passes iPad, says IDC.* 13 de Marzo de 2013.

[33]: Sencillo. RAE: Real Academia de la Lengua Española

[34]: Intuitivo. RAE: Real Academia de la Lengua Española

[35]: Google Design: Google Inc., *Tips for Android Design*, Android Developer, 2013.

ANEXO A: ACRÓNIMOS Y TÉRMINOS

PDA: *Personal Digital Assistant*. Dispositivo portátil capaz de realizar llamadas, así como facilitar el uso de aplicaciones tales como calendario, agenda, correo electrónico, bloc de notas, etc.

Smartphone: Teléfono inteligente. Terminal telefónico portátil (teléfono móvil) con un sistema operativo genérico capaz de hacer funcionar aplicaciones desarrolladas para uno de esos sistemas operativos (Android, iOS, etc.).

WI-FI: Estándar de red para la comunicación inalámbrica entre dispositivos.

API: *Application Programming Interface*. Conjunto de funciones o métodos que se incluyen dentro de una biblioteca de programación.

PPP: Puntos por Pulgada. Se trata de una medida de resolución, complementaria al tamaño de la pantalla y al número de píxeles de la misma.

Widget: O "artilugio". Son aplicaciones pequeñas que requieren de un motor de ejecución. Sirven para acceder de manera rápida a funcionalidades o aplicaciones del terminal. Ejemplos son: reloj, calculadora, linterna, agenda, mensajes,…

Kernel: Núcleo de un sistema operativo. Es la parte del sistema operativo, normalmente consistente en un gran fichero de varios millones de líneas de código que facilita a las aplicaciones y al propio sistema un acceso seguro al hardware.

Código Abierto: Se le conoce así al software que se distribuye y desarrolla de manera libre. Este concepto se centra en utilidades prácticas más que en usos económicos o cuestiones éticas.

SDK: *Software Development Kit*. Son paquetes utilizados para el desarrollo de aplicaciones o mejoras de una plataforma concreta. Normalmente, cuentan con todo lo necesario para empezar a desarrollar.

Tarjeta SD: *Secure Digital Card*. Es una tarjeta de memoria usada en dispositivos portátiles tales como cámaras fotográficas, videoconsolas o teléfonos móviles.

ROM: *Read Only Memory*. Medio de almacenamiento utilizado en computadores y dispositivos electrónicos. Únicamente permite la lectura de información y nunca la escritura. El mayor ejemplo de este tipo de memoria es el CD-ROM.

Firmware: Conjunto de instrucciones de muy bajo nivel, grabados en la memoria de un dispositivo o equipo. Estas instrucciones son las que serán llamadas por capas más altas (software) para la realización de tareas.

Podcast: Elemento de distribución de contenido multimedia de forma que se pueda distribuir de forma masiva a suscriptores mediante sistemas de radiodifusión RSS.

Walled Garden: Se trata de un determinado software o plataforma con un acceso restringido, con el fin de que sea lo más seguro posible.

Pixel: La unidad más pequeña de color uniforme que compone una imagen digital.

Framework: o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software.

Padding: Relleno de caracteres a un elemento criptográfico de forma que todos los elementos comunes (normalmente bloques de cifrado) tengan el mismo tamaño.

ANEXO B: MANUAL DE USUARIO

B.1. IDENTIFICARSE

Si es la primera vez que se ejecuta la aplicación, Image Cipher mostrará la siguiente pantalla:

ILUSTRACIÓN 60: PANTALLA DE BIENVENIDA

En ella, usted deberá pulsar sobre el botón “Registrarse” para registrarse como usuario en la aplicación. Una vez haya pulsado, aparecerá la siguiente pantalla en su smartphone o tableta:

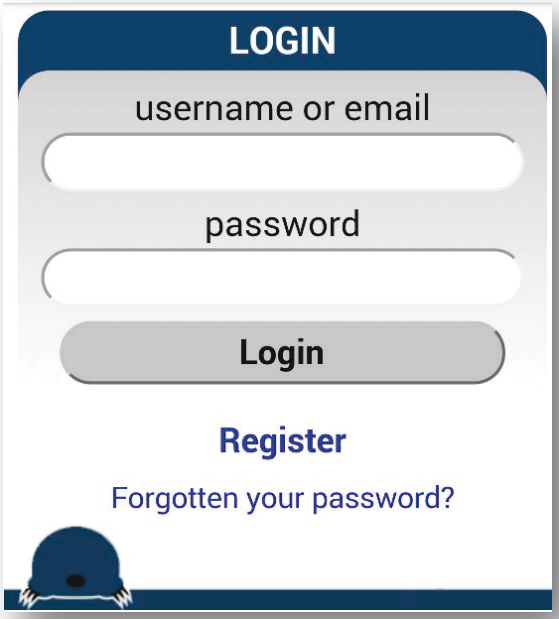


ILUSTRACIÓN 61: FORMULARIO DE AUTENTICACIÓN

Si usted no se ha registrado previamente, y tampoco cuenta con un registro anterior en una aplicación que utilice la plataforma Topoos, pulse sobre “Register” para acceder al formulario de registro. En caso contrario, introduzca su usuario y contraseña en el formulario. En caso de duda, trate de registrarse de nuevo.

La pantalla con el formulario de registro es la siguiente:

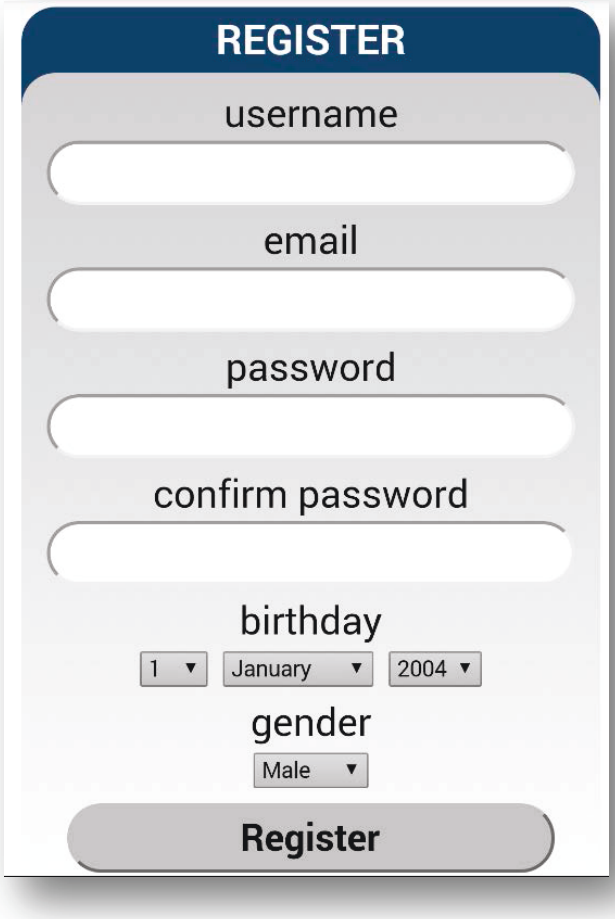


ILUSTRACIÓN 62: FORMULARIO DE REGISTRO

Para completar el registro complete los campos del formulario. Una vez acabado este proceso, si es correcto, redirigirá a la aplicación a la pantalla principal de la aplicación. EL proceso de registro sólo se hace una vez y la autenticación, salvo borrado de datos, se realizará de forma automática.

ILUSTRACIÓN 63: PANTALLA PRINCIPAL

La imagen anterior se corresponde con la pantalla principal de la aplicación Image Cipher. En ella usted podrá elegir entre cifrar una imagen, descifrarla o consultar las novedades de la versión.

B.2. CIFRAR UNA IMAGEN

Desde la pantalla principal de Image Cipher, pulse sobre el botón “Cifrar”. La aplicación la redirigirá a la pantalla siguiente:

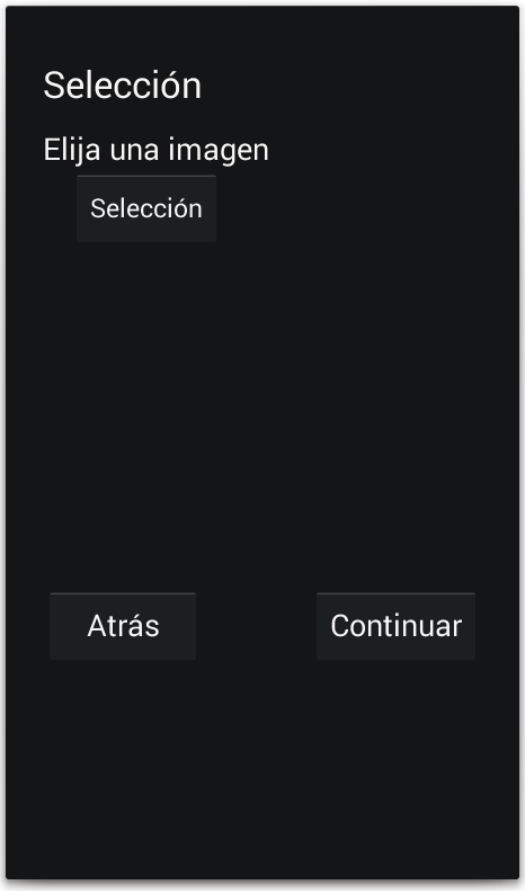


ILUSTRACIÓN 64: PANTALLA DE SELECCIÓN DE IMAGEN

Esta es la pantalla de selección de imágenes. Pulse sobre “Selección” para abrir la galería de Android y seleccionar la imagen que desee cifrar.

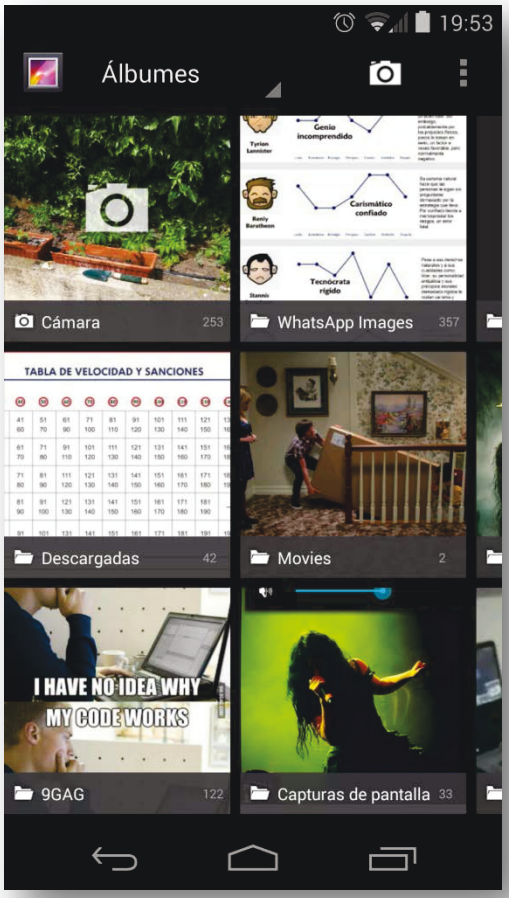


ILUSTRACIÓN 65: GALERÍA DE ANDROID

Seleccione una imagen cualquiera.

ILUSTRACIÓN 66: PANTALLA DE SELECCIÓN DE IMAGEN (IMAGEN SELECCIONADA)

Cuando haya seleccionado la imagen, ésta aparecerá en la pantalla de selección. Si usted ha seleccionado una imagen que no deseaba, puede volver a seleccionarla. Pulse “Continuar” para seguir el proceso de cifrado.

ILUSTRACIÓN 67: PANTALLA DE SELECCIÓN DE CONDICIONES

Una vez seleccionada la imagen, deberá elegir las condiciones que deberán cumplirse a la hora de descifrar la imagen. Esto se conoce como "métodos de cifrado". Pulse en los métodos que desee (Contraseña, WI-FI ID, Fecha / Hora y/o Usuario) para acceder a las pantallas correspondientes al método seleccionado.

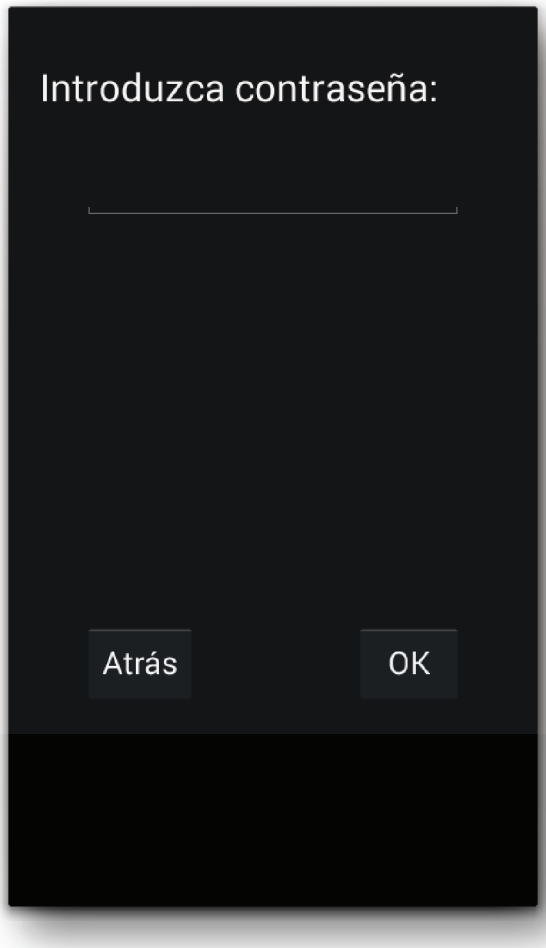


ILUSTRACIÓN 68: PANTALLA DE CONDICIÓN DE CONTRASEÑA

Si ha seleccionado “Contraseña” la aparecerá la imagen anterior. Introduzca una contraseña y pulse sobre “OK” para confirmar. Pulse “Atrás” para descartar.

ILUSTRACIÓN 69: PANTALLA DE CONDICIÓN DE WI-FI ID

Si pulsó sobre “WI-FI ID”, verá la imagen anterior. En ella usted puede añadir el método de cifrado de WI-FI ID de dos formas: Puede introducir el SSID de su punto de acceso a red manualmente o puede pulsar sobre “Identificar WI-FI”. SI usted desconoce el parámetro SSID, le recomendamos que pulse sobre “Identificar WI-FI”.

Si pulsa sobre “Identificar WI-FI”, la aplicación comprobará que su smartphone o tablet esté conectada a una red WI-FI, en caso contrario, Image Cipher le preguntará si desea conectar el dispositivo. Conecte el dispositivo si quiere que la aplicación identifique este método de acceso. Pulse en “OK” para continuar o en “Atrás” para descartar.

ILUSTRACIÓN 70: PANTALLA DE CONDICIÓN DE FECHA / HORA

Si ha seleccionado “Fecha / Hora”, Image Cipher mostrará la pantalla anterior. En ella usted podrá seleccionar la fecha y la hora a partir de la cual se podrá descifrar la imagen. Pulse “Seleccionar” para abrir los formularios de selección de ambos parámetros y pulse en “OK” para confirmar o en “Atrás” para descartar.

ILUSTRACIÓN 71: PANTALLA DE CONDICIÓN DE USUARIOS

La pantalla anterior se mostrará si usted pulsó sobre “Usuario” en la pantalla de métodos de cifrado. Simplemente introduzca el nombre de los usuarios que estarán autorizados a descifrar la imagen.

Una vez haya terminado de elegir los métodos de cifrado pulse en “Continuar”. Se le mostrará la pantalla siguiente, a modo de confirmación.

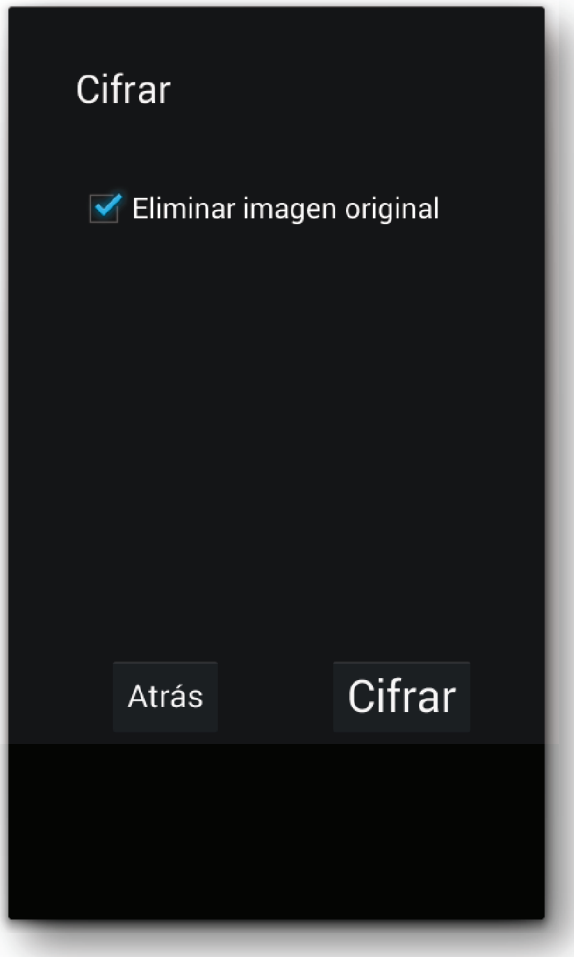


ILUSTRACIÓN 72: PANTALLA DE BORRADO DE ORIGINAL

En esta pantalla, usted tendrá la opción de eliminar la imagen original cuando concluya el proceso de cifrado. Si está seguro de haber seleccionado la imagen correcta con las condiciones de descifrado deseadas, pulse sobre “Cifrar” para comenzar el cifrado de la imagen.

ILUSTRACIÓN 73: PANTALLA DE PROGRESO (CIFRAR)

El progreso del cifrado se mostrará entonces en esta pantalla. Cuando concluya, se mostrará la pantalla de confirmación, que se describe en los apartados B.4. y B.5. de este manual.

B.3. DESCIFRAR UNA IMAGEN

Para descifrar una imagen previamente cifrada, deberá pulsar sobre “Descifrar” en la pantalla principal de Image Cipher. Entonces la aplicación le mostrará la pantalla de selección de imágenes:

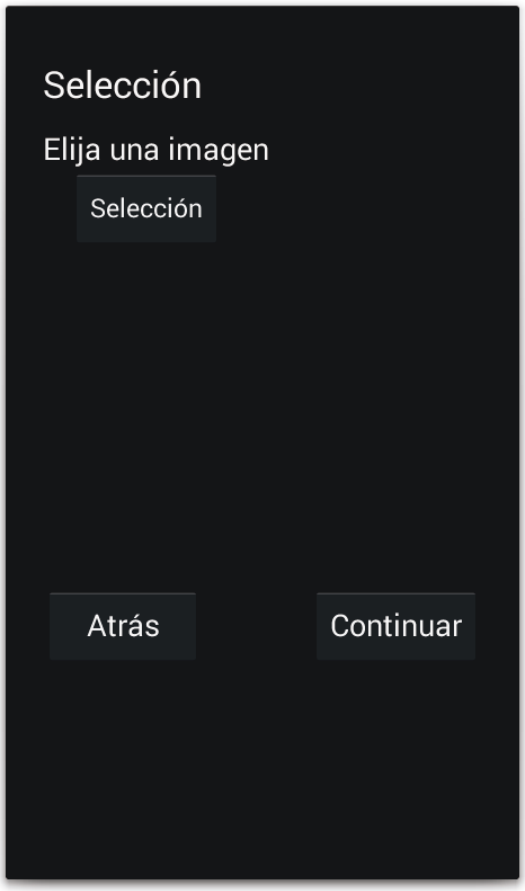


ILUSTRACIÓN 74: PANTALLA DE SELECCIÓN DE IMAGEN (DESCIFRAR)

En la pantalla deberá pulsar “Seleccionar” y elegir una imagen cifrada con Image Cipher. Pulse “Continuar” para seguir con el proceso. La aplicación comprobará si usted cumple con las condiciones necesarias para descifrar la imagen seleccionada.

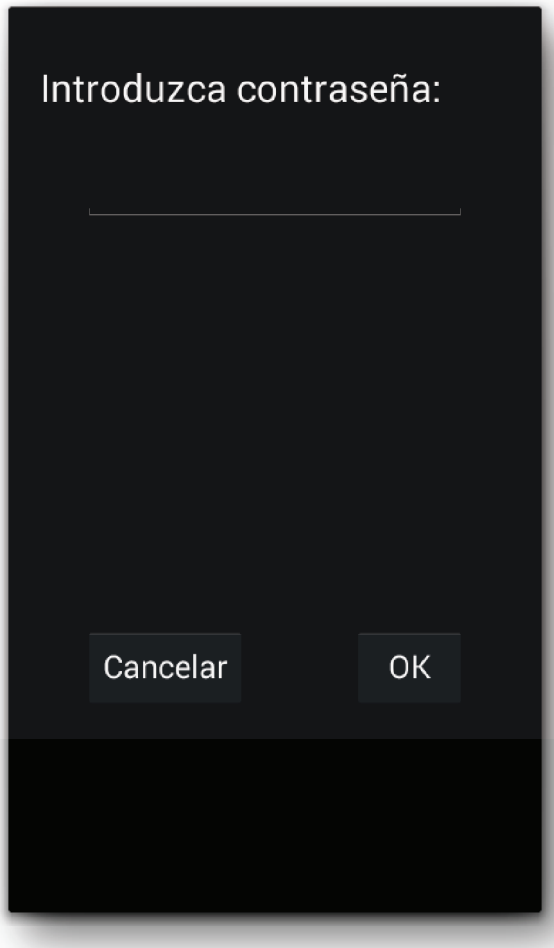


ILUSTRACIÓN 75: PANTALLA DE SOLICITUD DE CONTRASEÑA

En caso de que fuera necesaria contraseña, se le requerirá a usted en la pantalla anterior. Introdúzcala para continuar el descifrado de la imagen. Si todas las condiciones se cumplen, la aplicación pasará a descifrar la imagen y se mostrará la siguiente pantalla:

ILUSTRACIÓN 76: PANTALLA DE PROGRESO (DESCIFRAR)

En ella, se mostrará el progreso del descifrado. Una vez concluya, se mostrará la pantalla de confirmación, que se describe en los apartados B.4. y B.5. del presente manual.

B.4. ENVIAR UNA IMAGEN

Para enviar una imagen después de cifrarla, puede hacerlo desde la pantalla de confirmación, que para el caso del cifrado es la siguiente:

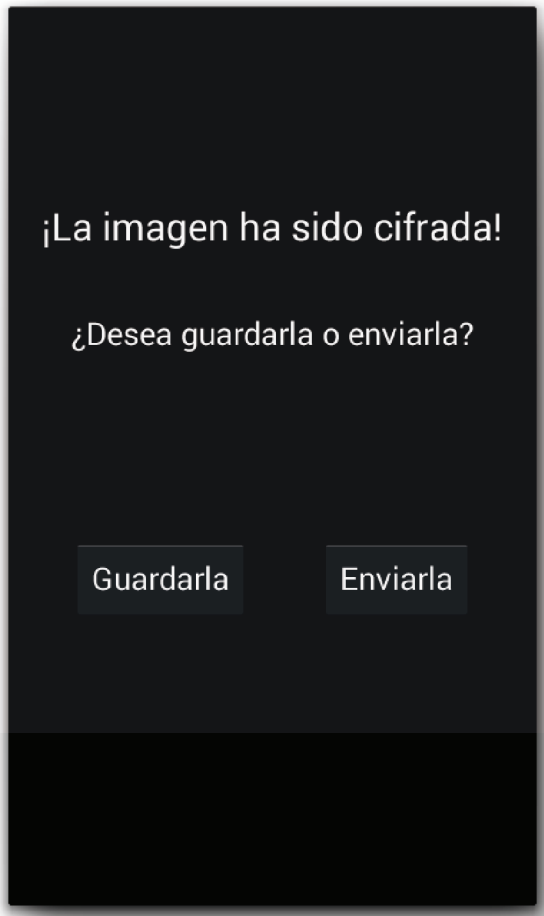


ILUSTRACIÓN 77: PANTALLA DE CONFIRMACIÓN (CIFRAR)

Para enviar la imagen cifrada, pulse en "Enviar" a continuación se mostrará un selector con las aplicaciones compatibles, capaces de enviar o compartir la imagen. Seleccione la aplicación que desee y confirme el envío. Image Cipher no permitirá el envío de imágenes descifradas.

B.5. ALMACENAR UNA IMAGEN

Image Cipher facilita el almacenamiento de imágenes a través de la pantalla de confirmación. En el caso de que se haya realizado un proceso de cifrado, la pantalla será la que se muestra en el apartado B.4. Si por el contrario se acaba de descifrar una imagen, la pantalla de confirmación será la siguiente:

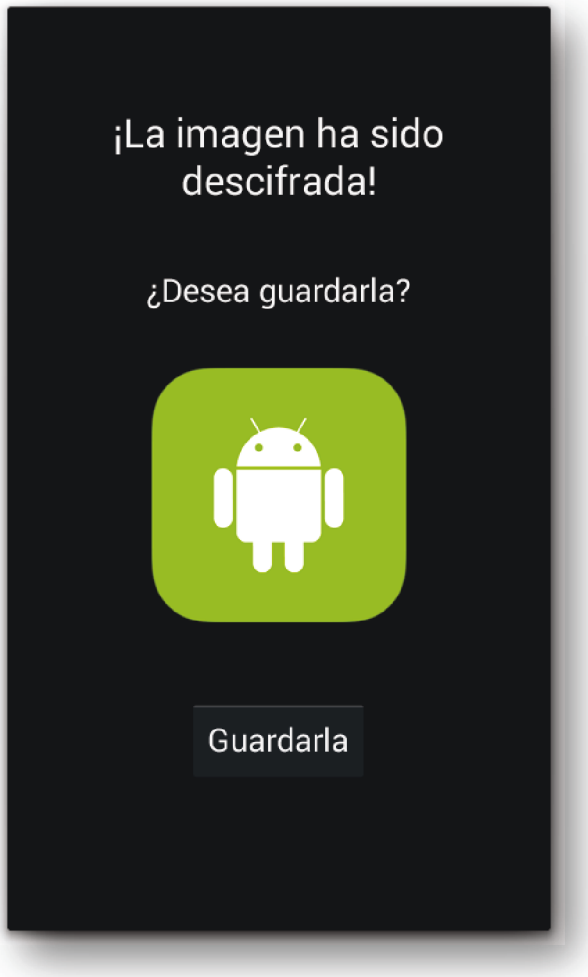


ILUSTRACIÓN 78: PANTALLA DE CONFIRMACIÓN (DESCIFRAR)

En ambos casos, para almacenar una imagen deberá pulsar en “Guardarla”. En el caso de la imagen descifrada, se almacenará en una carpeta a la que la galería de Android no tendrá acceso, de forma que no sea accesible tan fácilmente por otras aplicaciones.

B.6.VER NOVEDADES DE LA VERSIÓN

Para consultar las novedades de la versión de Image Cipher instalada en su dispositivo, sencillamente pulse sobre el número de la versión que aparece en la zona inferior derecha de la pantalla principal de la aplicación. Esto hará que la aplicación muestre una pantalla similar a la siguiente, donde se enumeran dichas novedades.

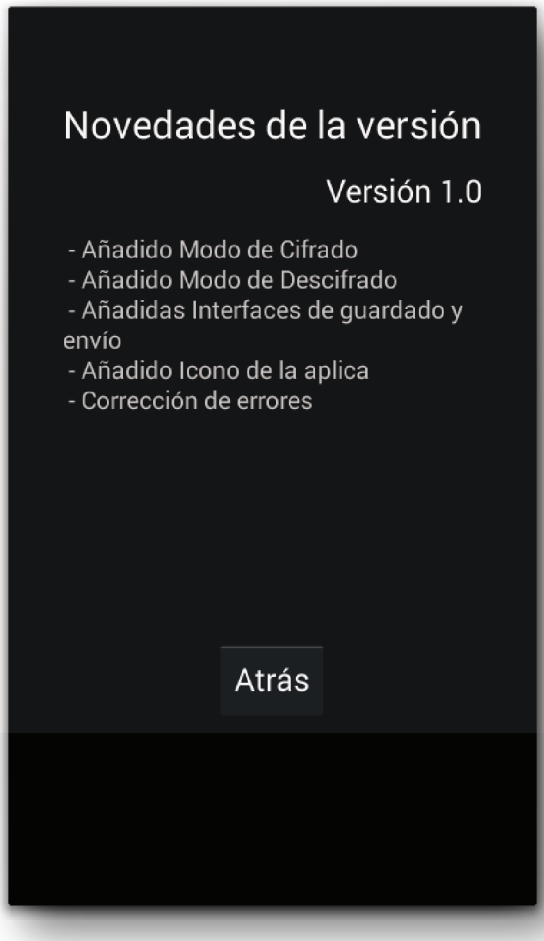


ILUSTRACIÓN 79: PANTALLA DE NOVEDADES

ANEXO C: USOS ÚTILES PARA LA APLICACIÓN

Como se comentaba en el estado actual del arte, una aplicación de cifrado de imágenes puede tener muchas posibilidades de uso. Este anexo, trata de explicar las más relevantes.

ENVÍO PRIVADO DE IMÁGENES

Image Cipher aporta seguridad protegiendo las imágenes que trata. El hecho de enviar una imagen con un cifrado "irrompible" garantiza un mínimo de seguridad que no aporta el envío de imágenes en claro. A nivel de usuario personal, esto es una gran ventaja, pero también encontramos con que empresas de todas las envergaduras también salen beneficiadas. Un claro ejemplo de ello es el creciente número de terminales utilizados. Estos móviles de empresa a menudo transportan información muy sensible, sobre todo se tratan de correos electrónicos o la agenda de contactos, que dependiendo de los mismos podemos considerar más o menos sensible desde el punto de vista de la seguridad.

Las imágenes cada vez contienen más información importante y cada vez con mayor grado de privacidad. El contenido de las mismas, no solo puede contener rostros, sino, capturas de pantalla con textos privados, tales como sms de empresas, resguardos bancarios etc.

Es re-marcadamente importante proteger este tipo de información y con las aplicaciones que actualmente se encuentran en el mercado, apenas puede hacerse.

ALMACENAMIENTO PRIVADO DE IMÁGENES

Este apartado está estrechamente relacionado con el anterior. Ya sabiendo la importancia de los datos que se manejan en estos casos, el almacenamiento debe ser igual de seguro que el envío. Los datos sensibles, tanto los almacenados en forma de imagen como los que no, pocas veces son eliminados, comprometiendo la seguridad del dispositivo que los almacena. El almacenamiento cifrado, con un algoritmo suficientemente fuerte puede salvarnos de una muy mala situación en caso de pérdida o robo, sobre todo si se manejan datos empresariales o bancarios, o incluso billetes de tren o de avión, que también cuentan con datos personales.

SORTEOS Y RIFAS

Una de las capacidades de Image Cipher es la de poder encriptar imágenes con condiciones de fecha y lugar. Esto puede explotarse en concursos o sorteos. Por ejemplo, si ponemos el caso de que una asociación o empresa sortea una determinada cantidad entre los participantes, los participantes deberán descargar la aplicación y registrarse en ella. Por otro lado, la "mano inocente" genera una imagen en la que indica que el usuario ha ganado el premio. La imagen se cifra con la condición de que solo esté disponible a partir de la hora en la que se reparte el premio, únicamente en el lugar donde se realice y naturalmente, solo podrá ser descifrada por el futuro usuario premiado.

Aunque es un uso poco convencional para una aplicación de cifrado, Image Cipher resulta estar perfectamente adaptada para este fin, como indica el ejemplo.

CUPONES DESCUENTO

Los cupones de descuento también son un uso posible para la aplicación desarrollada. Cada vez se utiliza menos el papel para usos a corto plazo. Una imagen en el teléfono móvil ya es válida como comprobante en multitud de casos ya es normal entrar en un avión sin el billete, si está almacenado en el terminal. Lo mismo pasará con los cupones descuento. Varias empresas de supermercados y de comida rápida utilizan la aplicación desarrollada para su empresa que envía los cupones cada vez que se arranca.

Sin embargo, esto implica un consumo de datos y de rendimiento. Si una empresa determinada utiliza Image Cipher para usar como cupones descuento, eliminaría esa restricción. Además limitaría los cupones a determinados usuarios (de forma que un usuario no tuviera el mismo descuento que otro) o sólo hacerlos efectivos en un cierto centro autorizado (mediante el WI-FI SSID) o a partir de determinada fecha.

ANEXO D: PLANIFICACIÓN Y PRESUPUESTO

PLANIFICACIÓN INICIAL

En este primer apartado se mostrará la planificación inicial que se hizo al comenzar el proyecto. Se incluye un diagrama de Gantt detallado:

Task Mode	Task Name	Durat	Start	Finish
	Fase Inicial	**6 days**	**Mon 13/01/14**	**Mon 20/01/14**
	Proposición del Proyecto	1 day	Mon 13/01/14	Mon 13/01/14
	Definición de Objetivos	3 days	Tue 14/01/14	Thu 16/01/14
	Planificación	2 days	Fri 17/01/14	Mon 20/01/14
	Análisis	**22 days**	**Tue 21/01/14**	**Wed 19/02/14**
	Análisis de Frameworks	7 days	Tue 21/01/14	Wed 29/01/14
	Estado del Arte	5 days	Thu 30/01/14	Wed 05/02/14
	Definición del Sistema	2 days	Thu 06/02/14	Fri 07/02/14
	Análisis de Requisitos	7 days	Sat 08/02/14	Mon 17/02/14
	Casos de Uso	2 days	Tue 18/02/14	Wed 19/02/14
	Diseño	**23 days**	**Thu 20/02/14**	**Mon 24/03/14**
	Diseño de la Arquitectura	2 days	Thu 20/02/14	Fri 21/02/14
	Diseño Módulo Cifrar	5 days	Mon 24/02/14	Fri 28/02/14
	Diseño Módulo Descifrar	5 days	Mon 03/03/14	Fri 07/03/14
	Diseño Pantalla Confirmación	5 days	Mon 10/03/14	Fri 14/03/14
	Diseño de las Interfaces de Usuario	6 days	Mon 17/03/14	Mon 24/03/14
	Implementación	**23 days**	**Tue 25/03/14**	**Thu 24/04/14**
	Módulo Cifrado	2 days	Tue 25/03/14	Wed 26/03/14
	Sistema de Cifrado	7 days	Thu 27/03/14	Fri 04/04/14
	Módulo Descifrado	2 days	Mon 07/04/14	Tue 08/04/14
	Sistema de Descifrado	5 days	Wed 09/04/14	Tue 15/04/14
	Almacenamiento	3 days	Wed 16/04/14	Fri 18/04/14
	Envío	2 days	Mon 21/04/14	Tue 22/04/14
	Resto de Módulos	2 days	Wed 23/04/14	Thu 24/04/14
	Evaluación y Pruebas	**8 days**	**Mon 28/04/14**	**Wed 07/05/14**
	Test de Funcionalidad	3 days	Mon 28/04/14	Wed 30/04/14
	Pruebas de Rendimiento	2 days	Thu 01/05/14	Fri 02/05/14
	Test de Integridad	2 days	Mon 05/05/14	Tue 06/05/14
	Análisis de Seguridad	1 day	Wed 07/05/14	Wed 07/05/14
	Fase Final	**24 days**	**Thu 08/05/14**	**Tue 10/06/14**
	Conclusiones	1 day	Thu 08/05/14	Thu 08/05/14
	Presupuesto	2 days	Mon 12/05/14	Tue 13/05/14
	Documentación	20 days	Wed 14/05/14	Tue 10/06/14

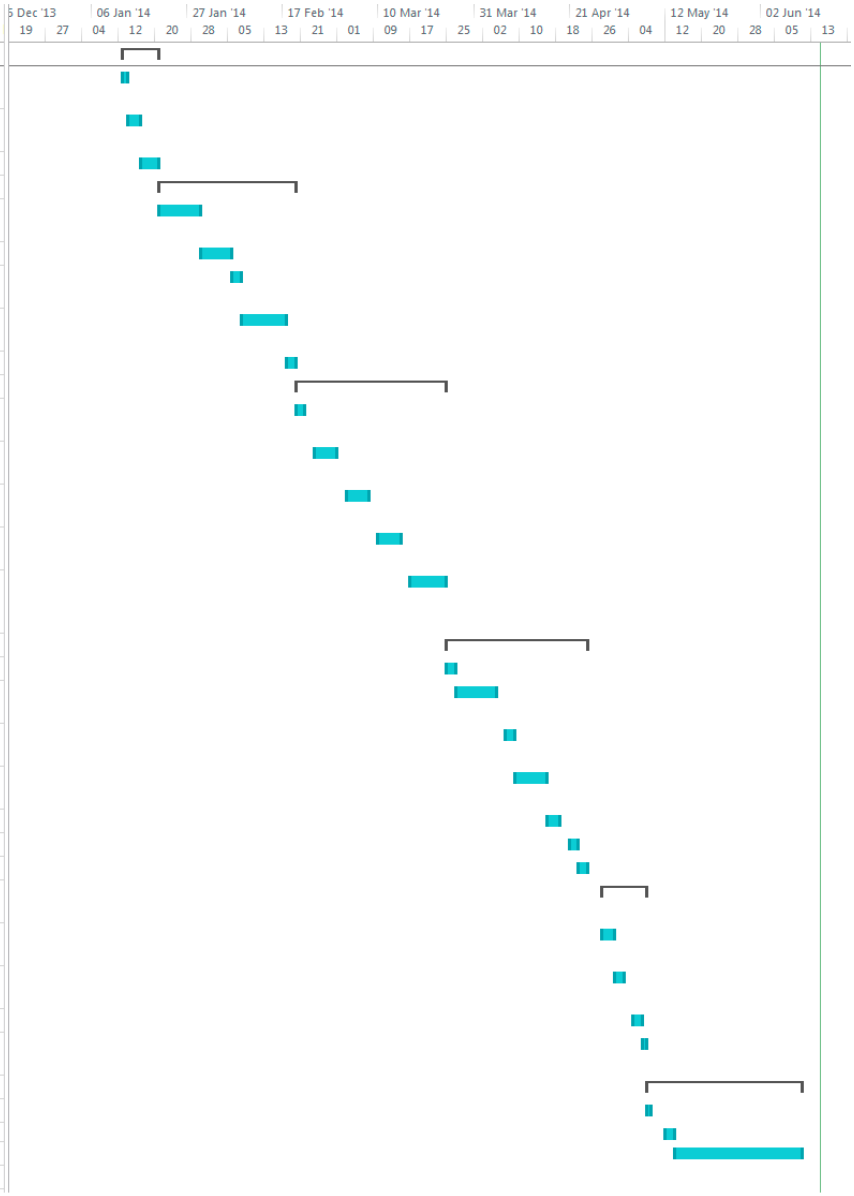
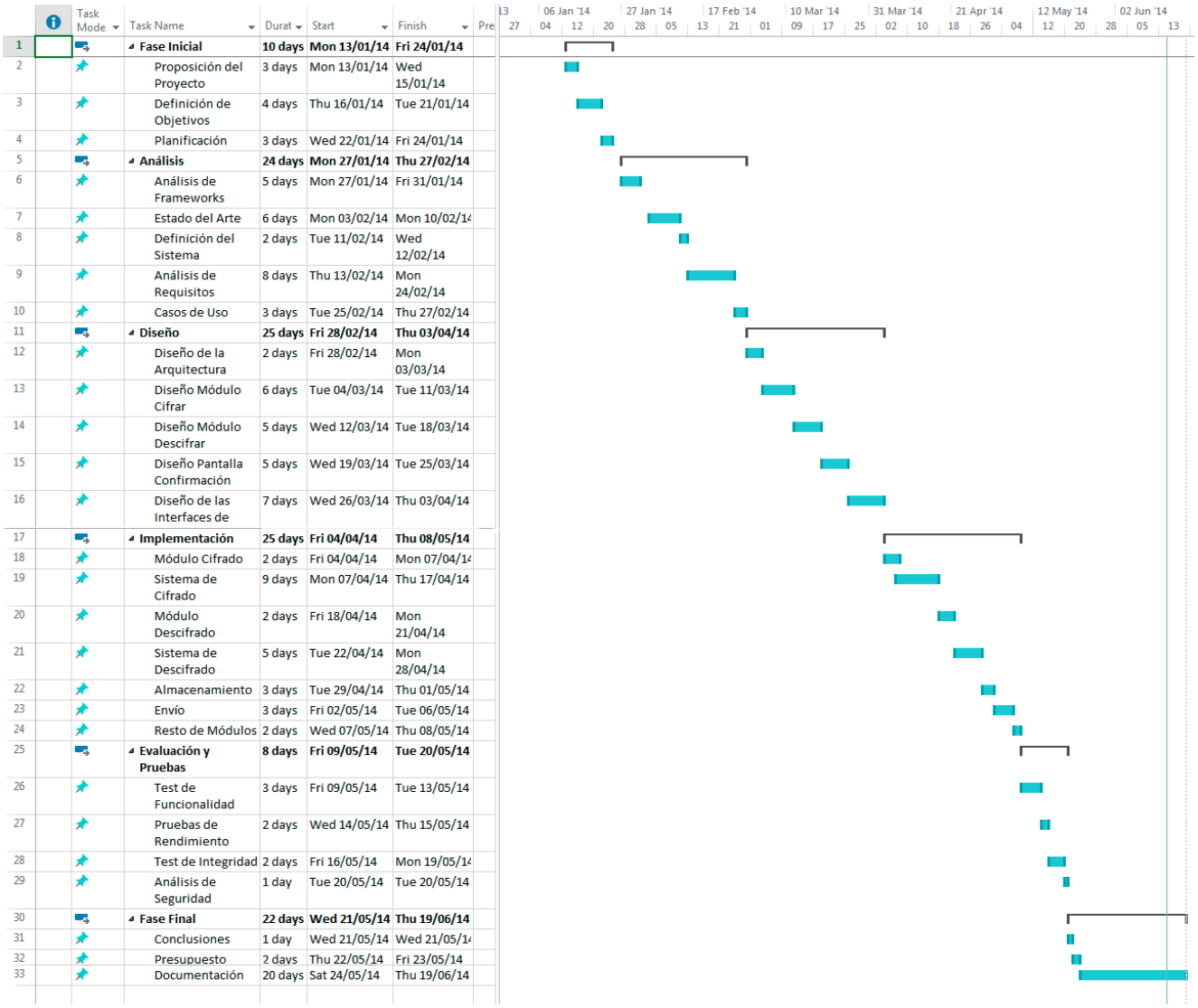


ILUSTRACIÓN 80: PLANIFICACIÓN INICIAL

PLANIFICACIÓN FINAL

En este apartado se muestra la planificación real, es decir, la que realmente se ha seguido con el desarrollo del proyecto:



	Task Mode	Task Name	Durat	Start	Finish
1		**Fase Inicial**	**10 days**	**Mon 13/01/14**	**Fri 24/01/14**
2		Proposición del Proyecto	3 days	Mon 13/01/14	Wed 15/01/14
3		Definición de Objetivos	4 days	Thu 16/01/14	Tue 21/01/14
4		Planificación	3 days	Wed 22/01/14	Fri 24/01/14
5		**Análisis**	**24 days**	**Mon 27/01/14**	**Thu 27/02/14**
6		Análisis de Frameworks	5 days	Mon 27/01/14	Fri 31/01/14
7		Estado del Arte	6 days	Mon 03/02/14	Mon 10/02/14
8		Definición del Sistema	2 days	Tue 11/02/14	Wed 12/02/14
9		Análisis de Requisitos	8 days	Thu 13/02/14	Mon 24/02/14
10		Casos de Uso	3 days	Tue 25/02/14	Thu 27/02/14
11		**Diseño**	**25 days**	**Fri 28/02/14**	**Thu 03/04/14**
12		Diseño de la Arquitectura	2 days	Fri 28/02/14	Mon 03/03/14
13		Diseño Módulo Cifrar	6 days	Tue 04/03/14	Tue 11/03/14
14		Diseño Módulo Descifrar	5 days	Wed 12/03/14	Tue 18/03/14
15		Diseño Pantalla Confirmación	5 days	Wed 19/03/14	Tue 25/03/14
16		Diseño de las Interfaces de	7 days	Wed 26/03/14	Thu 03/04/14
17		**Implementación**	**25 days**	**Fri 04/04/14**	**Thu 08/05/14**
18		Módulo Cifrado	2 days	Fri 04/04/14	Mon 07/04/14
19		Sistema de Cifrado	9 days	Mon 07/04/14	Thu 17/04/14
20		Módulo Descifrado	2 days	Fri 18/04/14	Mon 21/04/14
21		Sistema de Descifrado	5 days	Tue 22/04/14	Mon 28/04/14
22		Almacenamiento	3 days	Tue 29/04/14	Thu 01/05/14
23		Envío	3 days	Fri 02/05/14	Tue 06/05/14
24		Resto de Módulos	2 days	Wed 07/05/14	Thu 08/05/14
25		**Evaluación y Pruebas**	**8 days**	**Fri 09/05/14**	**Tue 20/05/14**
26		Test de Funcionalidad	3 days	Fri 09/05/14	Tue 13/05/14
27		Pruebas de Rendimiento	2 days	Wed 14/05/14	Thu 15/05/14
28		Test de Integridad	2 days	Fri 16/05/14	Mon 19/05/14
29		Análisis de Seguridad	1 day	Tue 20/05/14	Tue 20/05/14
30		**Fase Final**	**22 days**	**Wed 21/05/14**	**Thu 19/06/14**
31		Conclusiones	1 day	Wed 21/05/14	Wed 21/05/14
32		Presupuesto	2 days	Thu 22/05/14	Fri 23/05/14
33		Documentación	20 days	Sat 24/05/14	Thu 19/06/14

ILUSTRACIÓN 81: PLANIFICACIÓN FINAL

PRESUPUESTO

1. **Autor:** Iñaki Rodríguez López
2. **Departamento:** Departamento de Informática
3. **Descripción del Proyecto:**
    a. **Título**: Desarrollo de una aplicación de cifrado de imágenes en el sistema Android.
    b. **Duración (meses):** 4
    c. **Tasa de Costes Indirectos:** 18%
4. **Presupuesto final (€):** 23510,62
5. **Desglose presupuestario:**

Personal					
Apellidos	**Nombre**	**Puesto**	**Horas de Trabajo**	**Coste (€/h.)**	**Coste Total (€)**
Rodríguez López	Iñaki	Responsable de Proyecto	88	25	2200
Rodríguez López	Iñaki	Analista	192	20	3840
Rodríguez López	Iñaki	Diseñador	200	18	3600
Rodríguez López	Iñaki	Programador	200	14	2800
Rodríguez López	Iñaki	Encargado de Pruebas	64	10	640
Rodríguez López	Iñaki	Documentalista	176	18	3168
				Total (€)	**16248**

TABLA 149: COSTES DE PERSONAL

Equipos				
Descripción	**Coste (€)**	**Dedicación (meses)**	**Período Depreciación**	**Coste Imputable (€)**
Samsung Galaxy Ace	100	4	60	5,5
Samsung Galaxy S III Mini	275	4	60	15,12
Portátil LG A-505	600	4	60	33
			Total (€)	**53,62**

TABLA 150: COSTES DE EQUIPOS

Software				
Descripción	**Coste (€)**	**Dedicación (meses)**	**Período Depreciación**	**Coste Imputable (€)**
Microsoft Office 2007 Proffesional	709	4	50	47
Microsoft Project 2010 Profesional	729	4	50	49
Adobe Photoshop CS4	300	4	50	20
			Total (€)	**116**

TABLA 151: COSTES DE SOFTWARE

Otros Costes:		
Descripción	**Empresa**	**Coste Imputable (€)**
Internet	Movistar	48
Luz Eléctica	Iberdrola	40
	Total (€)	**88**

TABLA 152: OTROS COSTES

6. Resumen de costes:

Concepto	Coste (€)
Personal	16248
Equipos	53,62
Software	116
Otros Costes	88
Costes Indirectos	2924,64
Total	**19430,26**
IVA (21%)	4080,3546
Total con IVA	**23510,6146**

TABLA 153: RESUMEN DE COSTES